AL/CF-TR-1995-0141

DESIGN STUDY FOR PASSIVE SENSOR ALTITUDE AND TERRAIN AWARENESS SYSTEM (U)

Adam X. Miao
Greg L. Zacharias

CHARLES RIVER ANALYTICS
55 WHEELER STREET
CAMBRIDGE, MA 02138

Richard Warren

CREW SYSTEMS DIRECTORATE
HUMAN ENGINEERING DIVISION
WRIGHT-PATTERSON AFB OH 45433-7022

JUNE 1995

FINAL REPORT FOR THE PERIOD 1 SEPTEMBER 1994 TO 31 AUGUST 1995

Approved for public release; distribution is unlimited

AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6573

NOTICES

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Please do not request copies of this report from the Armstrong Laboratory. Additional copies may be purchased from:

National Technical Information Service
5285 Port Royal Road
Springfield, Virginia 22161

Federal Government agencies and their contractors registered with the Defense Technical Information Center should direct requests for copies of this report to:

Defense Technical Information Center
8725 John J. Kingman Road, Suite 0944
Ft. Belvoir, Virginia 22060-6218

DISCLAIMER

This Technical Report is published as received and has not been edited by the Technical Editing Staff of the Armstrong Laboratory.

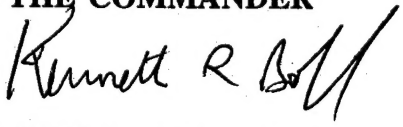
TECHNICAL REVIEW AND APPROVAL

AL/CF-TR-1995-0141

This report has been reviewed by the Office of Public Affairs (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER



KENNETH R. BOFF, Chief
Human Engineering Division
Armstrong Laboratory

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1995	Final Report 1 Sep 94 - 31 Aug 95

4. TITLE AND SUBTITLE

Design Study for Passive Sensor Altitude and Terrain Awareness System

5. FUNDING NUMBERS

C: F41624-92-C-6007
PE: 62202F
PR: 3005
TA: H2
WU: 08

6. AUTHOR(S)

Adam X. Miao
Greg L. Zacharias

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Charles River Analytics
55 Wheeler St
Cambridge, MA 02138

8. PERFORMING ORGANIZATION REPORT NUMBER

Final Report # R92191

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Armstrong Laboratory, Crew Systems Directorate
Human Engineering Division
Human Systems Center
Air Force Materiel Command
Wright-Patterson AFB OH 45433-7022

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

AL/CF-TR-1995-0141

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This Phase II effort completed the successful enhancement, implementation, and demonstration of the passive sensor Altitude and Terrain Awareness System (ATAS) operation using both computer generated and real imagery. The products of this effort include: 1) a test bed and an overall architecture of ATAS that comprises both the real-time image generation and real-time egomotion and terrain shape estimation; and 2) hardware and software specifications for a real-time flight rated ATAS. The test bed is composed of a flat-terrain image generator that produces CGI frames simulating perspective images viewed by an observer (camera) flying over flat terrain decorated by arbitary sinusoidal patterns, a rolling-terrain image generator that produces video (analog) imagery simulating images viewed by an observer flying over a sinusoidal rolling terrain decorated by sinusoidal patterns, and a frame grabber using hardware and software image grabbing tools hosted by a Macintosh Quadra 840 AV computer. ATAS is composed of an image processor; a snapshot egomotion and terrain shape estimator; and a Kalman filter, to provide smoother high-accuracy on-line estimates. Furthermore, we developed two types of egomotion and terrain shape estimators to deal with different kinds of terrain conditions: a geometry-based estimator and a structure-based estimator. We conducted a successful on-line demonstration of the ATAS ground-based prototype, driving the system with CGI and flight-recorded imagery, generating running estimates of altitude and attitude with respect to the overflown terrain.

14. SUBJECT TERMS

Passive Navigation, Optic Flow, Terrain Following/Terrain Avoidance
Altitude Estimation

15. NUMBER OF PAGES

118

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED

This page intentionally left blank.

Abstract

This Phase II effort completed the successful enhancement, implementation, and demonstration of the passive sensor Altitude and Terrain Awareness System (ATAS) operation using both computer generated and real imagery. The products of this effort include: 1) a test bed and an overall architecture of ATAS that comprises both the real-time image generation and real-time egomotion and terrain shape estimation; and 2) hardware and software specifications for a real-time flight rated ATAS. The test bed is composed of a flat-terrain image generator that produces CGI frames simulating perspective images viewed by an observer (camera) flying over flat terrain decorated by arbitrary sinusoidal patterns, a rolling-terrain image generator that produces video (analog) imagery simulating images viewed by an observer flying over a sinusoidal rolling terrain decorated by sinusoidal patterns, and a frame grabber using hardware and software image grabbing tools hosted by a Macintosh Quadra 840 AV computer. ATAS is composed of an image processor; a snapshot egomotion and terrain shape estimator; and a Kalman filter, to provide smoother high-accuracy on-line estimates. Furthermore, we developed two types of egomotion and terrain shape estimators to deal with different kinds of terrain conditions: a geometry-based estimator and a structure-based estimator. We conducted a successful on-line demonstration of the ATAS ground-based prototype, driving the system with CGI and flight-recorded imagery, generating running estimates of altitude and attitude with respect to the overflown terrain. The evaluation results indicate that ATAS has a capability for accurate, efficient, and robust egomotion and terrain shape estimation over various terrain (flat and non-flat) and under different motion conditions, when the proper estimators (the geometry-based estimator or the structure-based estimator) and the proper pixellation sizes are selected. The Phase II findings demonstrated the overall capability of ATAS and established benchmark criteria for evaluating future systems. The effort demonstrated operation with off-the-shelf hardware, at performance levels that can significantly enhance critical on-board functions, including passive navigation, optically-based TF/TA, and FLIR-based target designation and weapons control.

Significant commercial potential exists for ATAS, both in military and non-military applications. In the military applications, the passive motion and terrain shape estimation capability provided by ATAS can offer significant improvement to pilot's TF/TA subsystems, can drive helmet-mounted displays for pilot terrain awareness, and can enhance stealth operation. In the non-military applications, the same passive motion and terrain shape estimation capability can play even a larger role in the development of the robot navigation/guidance, automatic vehicle collision avoidance, and virtual reality systems.

Acknowledgment

This work was performed under USAF Contract F41624-92-C-6007. The authors thank the Technical Monitor, Dr. Rik Warren of USAF Armstrong Laboratory for his support and direction on this project, Dr. Ed Riley of Charles River Analytics for his technical assistance in both the completion of this contract and the creation of this document, and Ms. Kristy Shriver and Ms. April Blumenstiel for their help in creation and editing of this document. The authors also thank Mr. Phillip N. Smith and Mr. C. Thomas Synder of NASA Ames Research Center for letting us share their image data base.

Table of Contents

1. INTRODUCTION 1
1.1 Technical Objectives 3
1.2 Technical Approach 4
1.3 Summary of Results 5
1.4 Report Outline 9
2. Literature Review and Enabling Technologies 10
2.1 Review of Computer Vision Approach to Flow Based Motion and Terrain Estimation 10
2.2 Enabling Technologies 12
2.2.1 Kalman Filter Based Direct Approach to Egomotion and Terrain Shape Estimation 13
2.2.2 Singular Value Decomposition (SVD) Least Squares (LS) Estimation Algorithm 14
2.2.3 Square-Root Kalman Filter Algorithms 17
3. Passive Sensor Motion and Terrain Awareness System (ATAS) 20
3.1 System Architecture 20
3.2 Problem Formulation and Estimation Constraint Equation 21
3.3 Image Processor 23
3.4 Snapshot Estimator 25
3.4.1 Geometry Based Snapshot Estimator 26
3.4.2 Structure-Based Snapshot Estimator 29
3.5 Kalman Filter 34
3.5.1 Motion State Kalman Filter 35
3.5.2 Terrain Impact Time Map Kalman Filter 37
4. System Demonstration and Evaluation 41
4.1. Image Generation and Performance Criterion 41
4.1.1 CGI Generation 42
4.1.2 Real Imagery Generation 47
4.1.3 Frame Grabber 48
4.1.4 Performance Criterion 50
4.2 Evaluation Using Flat Terrain CGI 50
4.2.1 Estimation Accuracy with Flat Terrain CGI 51
4.2.2 Structure Based Estimator 56
4.2.2 Comparison of ATAS Predictions with Human Experiments 59
4.2.2.2 Comparison with Grunwald and Kohn's Experiment 63
4.3 Evaluation Using Non-Flat Terrain CGI 65
4.4. Evaluation Using Real Imagery 69
4.4.1 Geometry-Based Estimator 69
4.4.2 Structure-Based Estimator 77
4.5 Summary 88
5 Summary, Conclusions, and Recommendations 90
5.1 Summary 90
5.2 Conclusions 91
5.3 Recommendations 94
6. References
..... 10
0

List of Figures

Figure 1-1: Image Flow Associated with a Dynamically Changing Image 2
Figure 1-2: System Concept for Passive Sensor Altitude and Terrain Awareness System (ATAS) 3
Figure 1-3: A Conventional ATAS Architecture 3
Figure 2.2-1: Egomotion and Terrain Shape Estimation System 13
Figure 2.2-2: Kalman Filter Block Diagram 18
Figure 3.1-1: Passive Sensor Altitude and Terrain Awareness System 20
Figure 3.2-1: Line-of-Sight Rate 21
Figure 3.2-2: Image Plane Geometry 22
Figure 3.4-1: Geometry Baed Approach to Snapshot Estimation 26
Figure 3.4-2: Geometry-Based Snapshot Estimator 28
Figure 3.4-4: Problem Formulation of Structure Extraction 31
Figure 3.4-3: Structure Extraction Based Snapshot Estimator 33
Figure 3.5-1: Kalman Filter 34
Figure 3.5-2: Observer Motion Changes 39
Figure 3.5-3: Image Coordinates 40
Figure 4.1-1: Image Generator 41
Figure 4.1-2: Flat Terrain Image Generator 43
Figure 4.1-3a: High Resolution Image at Zero Depression Angle 44
Figure 4.1-3b: Low Resolution Image at Zero Depression Angle 44
Figure 4.1-3c: High Resolution Image at 30° Depression Angle 45
Figure 4.1-3d: Low Resolution Image at 30 Depression Angle 45
Figure 4.1-4a: High Resolution Image at 40° Depression Angle & 65° FOV 45
Figure 4.1-4b: Low Resolution Image at 30° Depression Angle & 65° FOV 45
Figure 4.1-5a: High Resolution Image at 45° Depression Angle & 50 ft Wavelength 46
Figure 4.1-5b: Low Resolution Image at 45° Depression Angle & 50 ft Wavelength 46
Figure 4.1-6: Rolling Terrain Image Generator 46
Figure 4.1-7: Rolling Terrain Images 47
Figure 4.1-8: An Image Frame from the Yosemite Sequence 47
Figure 4.1-10: Real-Time Image Generator Implementation 48
Figure 4.1-11a: Grabbed Image 49
Figure 4.1-11b: Ideally Generated Image 49
Figure 4.2-1: High Resolution Image 51
Figure 4.2-2: Low Resolution Image Used for Estimation 51
Figure 4.2-3: Impact Time Estimation History 52
Figure 4.2-4: Heading Angle Estimation History 53
Figure 4.2-5: Flight Path Angle Estimation History 53
Figure 4.2-6: Pitch Angular Velocity Estimation History 54
Figure 4.2-7: Impact Time Error History 54

Figure 4.2-8: Aimpoint Error History 55
Figure 4.2-9: Angular Rate Error History 55
Figure 4.2-10: Impact Time Estimation History 56
Figure 4.2-11: Heading Angle Estimation History 57
Figure 4.2-12: Flight Path Angle Estimation History 57
Figure 4.2-13: Pitch Angular Velocity Estimation History 58
Figure 4.2-14: Impact Time Error History 58
Figure 4.2-15: Aimpoint Error History 59
Figure 4.2-16: Angular Rate Error History 59
Figure 4.2-17: CGI Used for Simulating Warren Experiment 61
Figure 4.2-18: Heading Estimation Error Time History (Zero Image Noise) 61
Figure 4.2-19: Heading Estimation Error Time History (10% Image Noise) 62
Figure 4.2-20: Estimation Error vs. Perceptual Noise Level 63
Figure 4.2-21: CGI used for Simulating Grunwald and Kohn's Experiment 64
Figure 4.2-22: Heading Estimation Error Time History (15% Image Noise) 64
Figure 4.2-23: Heading Estimation Error vs. Velocity Height Ratio (Data vs. Model) 65
Figure 4.3-1: Single Frame of Yosemite Image Sequence 66
Figure 4.3-2: The Third Impact Time Map Generated from Yosemite Sequence 67
Figure 4.3-3: The Ninth Impact Time Map Generated from Yosemite Sequence 67
Figure 4.3-4: Aimpoint Error History 68
Figure 4.3-5: Angular Rate Error History 68
Figure 4.4-1a: First Image of NASA Sequence 69
Figure 4.4-2b: Last Image of NASA Sequence 69
Figure 4.4-2: Impact Time Estimation History 70
Figure 4.4-3: Heading Angle Estimation History 71
Figure 4.4-4: Flight Path Angle Estimation History 71
Figure 4.4-5a: Roll Angular Velocity Estimation History 71
Figure 4.4-5b: Pitch Angular Velocity Estimation History 72
Figure 4.4-5c: Yaw Angular Velocity Estimation History 72
Figure 4.4-6: Aimpoint Error History 72
Figure 4.4-7: Angular Rate Error History 73
Figure 4.4-8a: Superpixellated First Image of NASA Sequence 73
Figure 4.4-8b: Superpixellated Last Image of NASA Sequence 73
Figure 4.4-9: Impact Time Estimation History 74
Figure 4.4-10a: Heading Angle Estimation History 75
Figure 4.4-10b: Flight Path Angle Estimation History 75
Figure 4.4-11a: Roll Angular Velocity Estimation History 75
Figure 4.4-11b: Pitch Angular Velocity Estimation History 76
Figure 4.4-11c: Yaw Angular Velocity Estimation History 76
Figure 4.4-12: Impact Time Error History 76

Figure 4.4-13: Aimpoint Error History 77
Figure 4.4-14: Angular Rate Error History 77
Figure 4.4-15: Impact Time Estimation History 78
Figure 4.4-16a: The Fifth Impact Time Map 78
Figure 4.4-16b: The Fifteenth Impact Time Map 79
Figure 4.4-16c: The Twenty Fifth Impact Time Map 79
Figure 4.4-17a: Heading Angle Estimation History 80
Figure 4.4-17b: Flight Path Angle Estimation History 80
Figure 4.4-18a: Roll Angular Velocity Estimation History 81
Figure 4.4-18b: Pitch Angular Velocity Estimation History 81
Figure 4.4-18c: Yaw Angular Velocity Estimation History 82
Figure 4.4-19a: Aimpoint Error History 82
Figure 4.4-19b: Angular Rate Error History 83
Figure 4.4-20: Impact Time Estimation History 84
Figure 4.4-21a: Heading Angle Estimation History 84
Figure 4.4-21b: Flight Path Angle Estimation History 85
Figure 4.4-22a: Roll Angular Velocity Estimation History 85
Figure 4.4-22b: Pitch Angular Velocity Estimation History 86
Figure 4.4-23c: Yaw Angular Velocity Estimation History 86
Figure 4.4-24: Impact Time Error History 87
Figure 4.4-25: Aimpoint Error History 87
Figure 4.4-26: Angular Rate Error History 88
Figure 5.3-1: Architecture for Real-Time ATAS 95
Figure 5.3-2: Proposed Architecture for ATAS Flight Prototype System (FPS) 97
Figure B-1: Geometry of Projection Length Function B-2
Figure C-1: Projection Transform between Coordinate Systems C-1

List of Tables

Table 1.3-1: Comparison of Geometry-Based and Structure-Based Estimators 8
Table 2.1-1: Advantages and Disadvantages of Various Flow-Field Computation Methods 10
Table 5.3-1: ATAS Computational Requirements 95
Table 5.3-2: Estimated Costs for the Proposed Real-Time ATAS 96

Glossary of Abbreviations and Terms

ADC	Air Data Computer
ATAS	Altitude and Terrain Awareness System
CGI	Computer Generated Imagery
DMA	Defense Map Agency
DoD	Department of Defense
GBD	Ground-Based Demonstrator
FCP	Flight Computer Project
FLIR	Forward Looking Infra Red
FOV	Field Of View
FS	Flight System
GPS	Global Positioning System
INS	Inertial Navigation System
ISA	Industrial Standard Architecture
LOS	Line Of Sight
LS	Least Squares
NASA	National Aeronautics and Space Administration
SBIR	Small Business Innovation Research
SVD	Singular Value Decomposition
TF	Terrain Following
TF/TA	Terrain Following/Terrain Avoidance
TFR	Terrain Following Radar
TMB	Terrain Model Board

This page intentionally left blank.

1. INTRODUCTION

Terrain-following (TF) flight is performed when entering or leaving airspace in high threat battle areas. The intent is the timely movement towards an objective while using the terrain for masking. It combines vertical path control for following terrain contours with occasional lateral path control for heading changes and obstacle avoidance. This difficult and dangerous task demands a continuing awareness and control of one's own altitude, attitude, and velocity with respect to the terrain, and it results in high crew workload.

To help reduce this workload, we have designed and evaluated a passive sensor Altitude and Terrain Awareness System (ATAS), which, unlike conventional radar- or laser-based systems, operates with passively-sensed images to provide a real-time estimate of terrain-relative altitude. A system built around a passive imager rather than an active sensor significantly enhances covert operation and reduces countermeasure sensitivity. It has the potential for reducing the workload associated with the actual TF flight control and guidance tasks, without incurring an increased workload penalty in other areas, particularly in threat vigilance and defensive weapons control.

The TF environment imposes stringent requirements on any proposed passive-imager system. First, the system must be capable of operating with information-dense images of an ever-changing battlefield environment, images which are orders-of-magnitude more complex than what might be expected in, say, a multi-aircraft air-to-air engagement. Second, the system must be relatively robust with respect to image noise and be capable of operating in the presence of smoke, flares, and other incidental and deliberate sources which act to degrade the sensor image. Third, the system must operate in real-time at sample rates needed to drive the pilot's display at adequate rates. Finally, to ensure simplicity and reliability, the system must be relatively autonomous of other on-board systems, but capable of incorporating exogenous subsystem information available from sources such as the inertial navigation system (INS), an on-board Defense Map Agency data base, etc.

The system which can best satisfy all of these requirements is a passive imaging system which uses the feature-free optical flow-field present in dynamic image sequences, and illustrated in figure 1-1. This flow-field arises because of the relative motion of the aircraft with respect to the scene, causing the sensed image to flow with time, as the viewing geometry changes with time. Flow-fields are a function of the instantaneous viewing geometry (the location and orientation of the imager with respect to the visual world, the imager's field-of-view and boresight angle, etc.), the rate of change of this viewing geometry (the linear and angular velocity of the vehicle), and the intrinsic geometry of the viewed world. Flow-fields can be quite

complex, but the richness of the information contained in them makes them ideally suited to support a real-time TF ATAS using passively-sensed optical-flow.

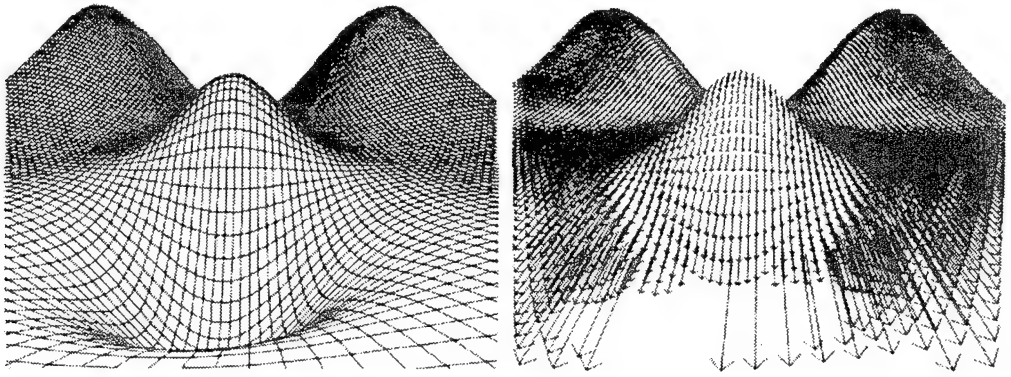


Figure 1-1: Image Flow Associated with a Dynamically Changing Image

The overall system concept of a passive ATAS operating in a low-level flight mode is illustrated in figure 1-2, and a conventional modular ATAS architecture is shown in figure 1-3. A forward-looking passive imager provides sequences of real-time two-dimensional dynamic images of the approaching terrain. A flow-field computer then generates the image flow arising from the aircraft's motion over the terrain. The computed image flow-field is then processed by a central estimator, which generates on-line estimates of the relevant aircraft states and terrain shape parameters. This latter set includes a three-dimensional impact time (depth) map that gives the flight time to all points in the imager field-of-view, where the impact time is defined as the speed-scaled range or depth. In effect, the system generates an instantaneous estimate of the topology and shape of the approaching terrain. This information is then fed to a terrain shape and altimeter module, which then determines terrain shape and terrain-relative altitude. These are then displayed to the pilot via a helmet-mounted display to provide continuous terrain awareness.

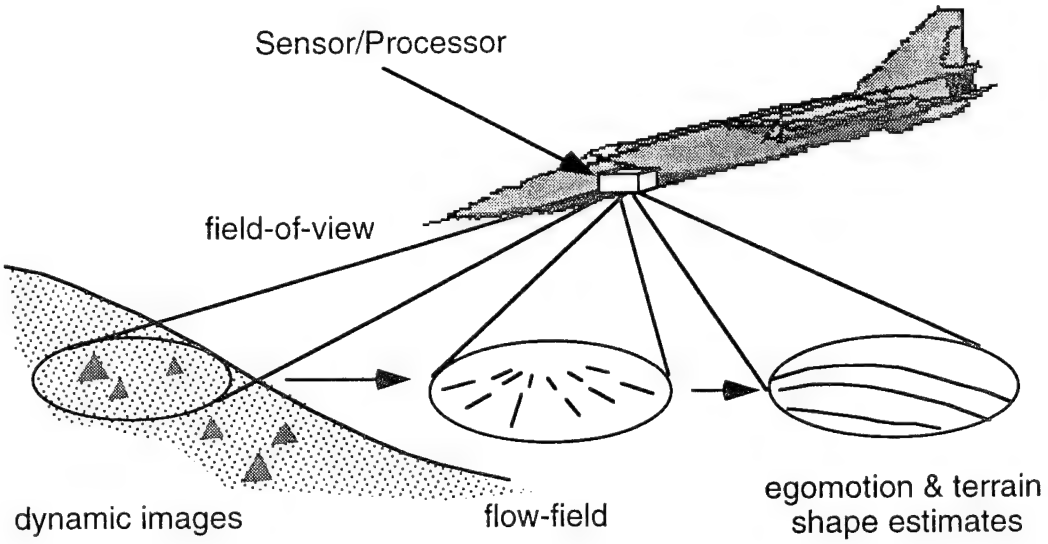


Figure 1-2: System Concept for Passive Sensor Altitude and Terrain Awareness System (ATAS)

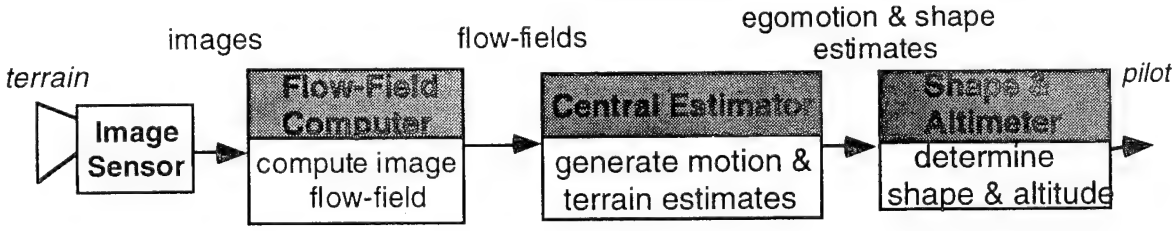


Figure 1-3: A Conventional ATAS Architecture

1.1 Technical Objectives

The primary objective of this Phase II SBIR effort was to develop and demonstrate a prototype version of the passive sensor Altitude and Terrain Awareness System (ATAS). Five goals were achieved:

- Enhancement of the Predecessor Phase I Design
- Implementation a ATAS Ground Based Demonstrator
- Development of a Test Bed
- Evaluation of the ATAS Ground Based Demonstrator
- Specification of Design Requirements for a Production System

We now describe these Phase II technical objectives in the following paragraphs.

The first objective of the Phase II effort was to enhance the Phase I ATAS design for

accurate, and robust egomotion and terrain shape estimation in realistic flight situations. The focus here was to upgrade the Phase I ATAS design—whose applicability was limited to flat terrain and constant motion—to situations involving non-flat terrain and non-constant motion as occurs in realistic TF situations.

The second objective of the Phase II effort was to implement a ground based demonstrator (GBD) for ATAS. The focus here was to ensure a modular and error-free implementation of ATAS in a conventional C programming language.

The third objective of the Phase II effort was to develop a test bed for ATAS. The focus here was to configure the hardware for image grabbing, to implement the computer image generators (CGI) for terrain image generation, and to develop performance metrics for system evaluation.

The fourth objective of the Phase II effort was to evaluate the ATAS ground-based demonstrator using the test bed developed. The focus here was the extensive evaluation using both the CGI and real imagery. Specifically, testing of the ATAS GBD was conducted using computed generated imagery from the test bed, and, most significantly, using flight-recorded and real-time frame-grabbed imagery from aircraft flying over actual terrain.

The third and last objective of the Phase II effort was to develop the design requirements for a production system to be developed under a Phase III effort. This included the specification of the hardware requirements and generation of the software specifications. The commercial application of this design requirement package would be a Phase III development effort to build a production system for a major avionics supplier to the air transport industry.

1.2 Technical Approach

Our Phase II technical approach to developing and demonstrating a prototype version of the passive sensor altitude and terrain awareness system (ATAS) was comprised of five tasks:

- Review of Current Approaches to Flow-Based Egomotion and Terrain Shape Estimation
- Enhancement and Development of a Prototype ATAS
- Development of a Test Bed for ATAS
- Demonstration and Evaluation of System Performance with CGI and flight-recorded imagery
- Recommendation of development path for a commercial product

We first **reviewed current approaches and evaluated their advantages and disadvantages**. A literature search was conducted specifically focusing on flow-field based

estimation technologies. A review and summary of the most promising studies were conducted to identify capabilities and limitations of earlier flow-field based approaches.

We then **designed and implemented a prototype ground based ATAS** via specifying its overall architecture, its functional modules, and its software implementation. Our efforts were focused on the development of a robust, accurate, and efficient ATAS that can work in real-time. A Kalman filter based direct approach was developed that comprised of three modules: 1) an image processor; 2) a snapshot egomotion and terrain shape estimator; and 3) a Kalman filter providing smoothed estimates.

We also developed and implemented a test bed that has a capability for flat and non-flat computer terrain image generation. We defined and implemented various performance metrics for system evaluation.

Following implementation of the ground based ATAS and its test bed, we **demonstrated and evaluated system performance** via an extensive testing program. ATAS was tested using both CGI and real imagery, and using both the flat and non-flat terrain imagery. It was also tested against human experiments.

Finally, we **recommended a development path** for a commercial product, based on the results of the evaluation effort. The development path includes an enhanced performance version of the Phase II design, and demonstration and performance evaluation over an extended CGI and real imagery database.

1.3 Summary of Results

The primary result of this Phase II effort was the successful enhancement, implementation, and demonstration of the ATAS operation using both computer generated and real imagery. The major findings supporting this effort can be summarized as follows.

We developed a test bed and an overall architecture of ATAS that comprises both the real-time image generation and real-time egomotion and terrain shape estimation. We developed and implemented a flat-terrain image generator that produces CGI frames simulating perspective images viewed by an observer (camera) flying over flat terrain decorated by arbitrary sinusoidal patterns. We also developed and implemented a rolling-terrain image generator that produces video (analog) imagery simulating images viewed by an observer flying over a sinusoidal rolling terrain decorated by sinusoidal patterns. In addition, from various sources we obtained other complex CGI terrain images used widely for computer vision algorithm testing. For real image generation, we obtained video imagery using a video camera fixed on a helicopter flying over real terrain. To convert the analog video imagery into digitized image frames, we implemented a

frame grabber using hardware and software image grabbing tools hosted by a Macintosh Quadra 840 AV computer.

For egomotion and terrain shape estimation, we developed a three stage modular architecture comprised of an image processor; a snapshot egomotion and terrain shape estimator; and a Kalman filter, to provide smoothed high-accuracy on-line estimates. This modular approach provided performance improvements and simplification for system implementation over the original design developed under the Phase I effort. First, it avoids the complicated interactions between low-level image processing and high-level motion state and terrain shape estimation. Second, separation of image processing, snapshot estimation, and Kalman filtering makes each problem simpler and its solution easier and more robust. Third, the system has a natural interface to incorporate external information for performance enhancement. For example, if the external information on observer motion state is available, it can then be easily incorporated into the Kalman filter without reconfiguration of the entire system. Finally, we believe that this modular architecture can provide the basis for a functional model of the human flow-field based vision process.

We developed two types of egomotion and terrain shape estimators to deal with different kinds of terrain conditions: a geometry-based estimator and a structure-based estimator. The geometry-based estimator assumes that the terrain shape satisfies a polynomial geometry, and uses this model for terrain shape estimation. The structure based estimator does not assume *a priori* a terrain model, but extracts the hidden structural relationship among imaged terrain points using a multi-layer terrain structure model.

For static estimation in the snapshot estimator and dynamic estimation in the Kalman filter, we employed and implemented two state-of-the-art estimation algorithms: the Singular Value Decomposition (SVD) based Least Squares (LS) algorithm and the Square-Root Kalman filter algorithm. The application of these two numerically robust algorithms effectively overcomes any numerical instabilities that might be present in the estimation process.

Furthermore, the operation of ATAS needs specification of only a few meaningful parameters. For the geometry based approach, only four parameters need to be specified: the terrain model, the initial heading vector, the model standard deviations for motion state dynamic equations, and the discount factor for terrain impact time (range) map evolution; for the structure based approach, only the latter two parameters need to be specified. In contrast, many more parameters would be needed in a flow or feature matching based system, many of which need to be determined by trial and error.

We conducted a successful on-line demonstration of the ATAS ground-based prototype,

driving the system with CGI and flight-recorded imagery, generating running estimates of altitude and attitude with respect to the overflown terrain. The evaluation results indicate that ATAS has a capability for accurate, efficient, and robust egomotion and terrain shape estimation over various terrain (flat and non-flat) and under different motion conditions, when the proper estimators (the geometry-based estimator or the structure-based estimator) and the proper pixellation sizes are selected.

Using CGI, we evaluated ATAS for both flat and non-flat terrain images. With the flat terrain, ATAS had estimation errors of less than 2 degree in aimpoint, less than 3 degree/second in angular rate, and less than 5% in for altitude (impact time).

With the non-flat terrain images, we used a complex sequence that was generated by simulating a flight over the Yosemite valley. ATAS demonstrated an excellent ability to recover the complicated Yosemite valley structure, whose complexity is far beyond the capability of any polynomial geometry terrain models (flat, quadratic, etc.). Moreover, ATAS had a less than 5 degree error in aimpoint, and a less than 1 degree/second error in angular rate, for these complicated non-flat terrain images.

For the flight-recorded and real-time grabbed terrain images, even though they were poorly "decorated", had very high noise-to-signal ratios, and the observer (airplane) was not undergoing constant motion, ATAS was still capable of generating accurate egomotion and terrain shape estimates. Specifically, it demonstrated errors of less than 5 degree in aimpoint, less than 2 degree/second in angular rate, and less than 10% in altitude (impact time). Moreover, ATAS showed an excellent ability in rapidly following and tracking the non-constant observer motion. Besides generating overall terrain shape estimates such as speed-scaled altitude, ATAS was also capable of recovering the shape of local objects (in this case, trucks present in the terrain images).

We also compared the advantages and disadvantages of the geometry-based and structure-based estimators, summarized in table 1.3-1.

The geometry-based estimator can only be applied to a terrain that can be approximated by a simple geometry model. Moreover, using a geometry-based model to approximate the entire terrain within the FOV makes it impossible to recover local structures hidden in the terrain image, such as isolated objects. In contrast, the structure-based approach has no limitation in its applicability. It can be applied to both simple and complicated terrain. Moreover, it can be used to recover the local structure of isolated objects in the imager FOV.

Large pixels can be used for image measurement computation in the geometry-based

estimator. Large pixels not only reduce the computation load greatly, but also enhance overall estimation accuracy. The structure-based estimator, however, requires small pixels to properly define the terrain structure and thus incurs additional computational costs.

The geometry-based estimator has superior performance over the structure-based estimator when the actual terrain can be approximated by a simple geometry. In this case, the geometry-based model represents a true description of the terrain shape being imaged. The structure-based estimator, however, always has an approximate representation of the actual terrain, no matter how simple or complicated it actually is.

Table 1.3-1: Comparison of Geometry-Based and Structure-Based Estimators

Terrain Model	Applicability	Pixellation Level	Performance
Geometry-Based	Simple geometry terrain	Works Better for large pixel	Better for simple geometry terrain
Structure-Based	Unlimited Can recover local structure	Requires small pixel	Better for complicated terrain or local structures

Using ATAS, we also analyzed several previous experimental studies of human egomotion perception. For the two reported on here, we used simple additive image noise to model the human's inability to register perfectly noise-free images, and we were able to show how this *perceptual noise* leads to corresponding inaccuracies in estimates of egomotion state. In particular, we showed how, within the model context, an assumed 10% to 15% image noise level leads directly to model-predicted egomotion aimpoint accuracy in the experimentally observed range of 1.5 to 2.5 degree. In addition, we showed how these aimpoint accuracy can be expected to co-vary with velocity-to-height ratios, which change by over a factor of 10, and confirmed these trends with experimental data showing the same trends. Although more model validation is clearly needed, we feel that these initial matches between model and data are very encouraging.

Finally, we identified requirements for production system development and demonstration. Specifically, we recommended that the Phase II laboratory prototype serve as the basis for the development of a flight prototype system, to be used to demonstrate in-flight capabilities and system potential. A two-step development and demonstration program is envisioned: 1) development and demonstration of a flight prototype system configured for in-flight test and evaluation; and 2) specification of design requirements for a follow-on production system.

The Phase II findings demonstrated the overall capability of ATAS and established benchmark criteria for evaluating future systems. The effort demonstrated operation with off-the-shelf hardware, at performance levels that can significantly enhance critical on-board functions, including passive navigation, optically-based TF/TA, and FLIR-based target designation and weapons control.

1.4 Report Outline

Chapter 2 provides technical background on the proposed passive sensor Altitude & Terrain Awareness System (ATAS). Section 2.1 reviews major flow-field computer vision approaches to the egomotion and terrain estimation problem, and outlines their advantages and disadvantages. Section 2.2 sketches our approach, and presents its enabling technologies.

Chapter 3 gives a technical description of ATAS. Section 3.1 presents an overview of the system and its two component modules: an image generator and an egomotion and terrain shape estimator. Section 3.2 then describes technical details of the image generator, while section 3.3 describes the egomotion and terrain shape estimator.

Chapter 4 demonstrates and evaluates overall operation of the ground-based ATAS. We show that ATAS can accurately generate navigation-critical state and terrain shape information, such as heading, angular velocity, and impact time (depth) map across the FOV. Section 4.1 describes the real-time implementation of the ATAS system and define the performance evaluation criterion. Section 4.2 presents evaluation results using flat terrain CGI, section 4.3 presents the evaluation results using non-flat terrain, and section 4.4 presents the evaluation results using flight-recorded imagery. Section 4.5 concludes the chapter with a summary of the evaluation results.

Chapter 5 concludes the report with a summary, conclusions, and recommendations for further development and demonstration.

2. LITERATURE REVIEW AND ENABLING TECHNOLOGIES

This chapter provides technical background on the proposed passive sensor Altitude and Terrain Awareness System (ATAS). Section 2.1 reviews major flow-field computer vision approaches to the egomotion and terrain estimation problem, and outlines their advantages and disadvantages. Section 2.2 sketches our approach, and presents its enabling technologies

2.1 Review of Computer Vision Approach to Flow Based Motion and Terrain Estimation

Computer vision approaches to flow based motion and terrain estimation problems typically formulate the problem as a common two-stage process: first, compute the optical *flow-field* based on the dynamically changing images; second, estimate the egomotion states and the relative depth/shape based on the computed flow and possibly on a model of the viewed terrain. The approach provides a convenient *divide and conquer* approach to egomotion and shape estimation. One group of researchers can thus concentrate on flow-field computation without considering its consequences for motion estimation, while another group can concentrate on the estimation of motion states and terrain shape, assuming availability of an accurately computed flow-field without worrying about the feasibility of its generation.

There are a number of algorithms for computing the optical flow-field, given a sequence of dynamically changing images. These algorithms can be roughly grouped into feature-based, gradient-based, and frequency-based approaches. Table 2.1-1 summarizes the advantages and disadvantages of each approach.

Table 2.1-1: Advantages and Disadvantages of Various Flow-Field Computation Methods

Method	Advantages	Disadvantages
Feature-based	Lowest computation load	Difficulty in matching features Sparse flow-field Not suitable for featureless imagery
Gradient-based	Easy implementation Wide applicability	Worst performance High computation load
Frequency-based	Best performance	Highest computation load A priori imagery information required Large number of frames

The feature-based approach extracts the flow-field by matching features from one image frame to the next and determining their shifts (Anandan, 1989; Singh, 1992; Sridhar & Chatterji, 1994). The approach has the advantage of the lowest computational requirement among the three approaches, since flow-fields are only computed for the sparse set of features in the image and shift computation involves only a few arithmetical operations. However, it has the disadvantage

of great difficulty in matching features from one image to the next due to uncertainties caused by occlusion, observer motion, and image sensor noise. Moreover, the approach provides only a sparse subset of the entire flow-field that depends on the number of recognizable features in an image, and cannot be used in featureless images.

The gradient-based approach computes the flow-field vector at each pixel from spatiotemporal gradients of (filtered or original) image intensity (Horn & Schunck, 1981; Nagel, 1983; Lucas & Kanade, 1981). The approach has the advantage of ease of implementation in both sequential and parallel computers since computation of spatiotemporal gradients at a pixel is simple and involves only a few neighbor pixels. The approach also has the advantage of applicability to both feature-rich and featureless images. However, it has the disadvantage of resulting in the worst performance in the case of non-smooth images due to occlusion boundaries and perceptual distortion. Furthermore, since the gradient-based approach usually must use an iterative computation method to improve the quality of the computed flow-field, the computational load for the approach can be high.

The frequency-based approach generates the flow-field vector using velocity-tuned filters in the Fourier domain (Heeger, 1987; Fleet, 1992; Fleet & Jepson, 1990). By using large numbers of images and rather stable frequency domain image features (energy or phase), the approach, in general, leads to the most accurate flow-field recovering. The approach, however, has the highest computational load, requires *a priori* motion and image information in designing the filters, and is the most difficult to implement since the filters must be tuned according to the image spectrum.

Real-time operation and lack of *a priori* motion and image (terrain) information limited our selection of flow-field computation algorithms to feature-based or gradient-based approaches. Existing flow-field based egomotion and terrain estimation technique requires a relative error in the computed flow-field to be less than 10% the reliable egomotion and terrain estimation according to Barron, Jepson & Tsotsos (1990). What can be achieved by current flow-field computation algorithms, however, is not encouraging. Barron, Fleet & Beauchemin (1994) conducted quantitative evaluation of three approaches. Results show that only the well-tuned frequency-based algorithms could achieve the required accuracy, whereas both the feature-based and gradient-based algorithms had average errors close to or above 10% even for synthetic (computer-generated) flat-terrain images. The same above 10% average errors were also reported in quantitative evaluation of the feature-based and gradient-based algorithms on two relatively simple synthetic image cases in Little & Verri (1989).

There are also a number of algorithms to estimate motion/shape parameters, given the

computed flow-field. These algorithms can be roughly grouped into instantaneous (snapshot) and Kalman filter estimators. Of the instantaneous estimators, Zacharias, Caglayan & Sinacori (1988) developed a quasi-static estimator to estimate an observer's instantaneous heading (vertical and lateral), angular velocity, and relative depth or impact *times* (the speed-scaled range or 3D shape) by decomposing the motion state and terrain shape estimation problems. Heeger & Jepson (1992) split the flow-field equation via algebraic manipulation into three sets of equations. The first set relates the flow field to only the translational (heading) motion. Thus, relative depth and rotation motion (angular velocity) need not be known prior to estimating the translational motion. Once the translational motion has been determined, the second set of equations relating to both translational and rotational motion is then used for estimating rotation. Finally, the depth at each pixel is estimated using the third set of equation, given the estimated translation and rotation.

Kalman filter based methods use the computed flow-field vectors as observations, and integrate them with the motion/shape parameters over multiple image frames to achieve robust estimation of motion/shape parameters. Various schemes have been developed by Broida & Chellappa (1986); Wu, Rink, Caelli & Gourishankar (1989); Sridhar & Chatterji (1994), and Matthies, Szelisky & Kanada (1994). Two fundamental problems here are: 1) the need to track and establish correspondence over a larger numbers of image points or features that appear/disappear or occlude each other; and 2) the numerical instability of the filtering process and the high computational cost due to the high dimensional nonlinear dynamics of the motion/shape parameters. These have restricted the application of Kalman filtering mostly to the use of feature-based flow-fields. Even in these cases, some promising results have been achieved in dealing with real images, but only when accurate motion states are known (Sridhar & Chatterji, 1994). Extension of the Kalman filter approach to non-feature based approaches was proposed by Matthies et al. (1994), but again, accurate motion states were also assumed to be known.

Noticing difficulties in accurately and efficiently estimating flow-fields, Negahdaripour (1986), Negahdaripour & Horn (1989), and Zacharias, Miao & Warren (1995) developed direct approaches that estimate egomotion and terrain shape *without* computing the flow-field. These direct approaches, however, only consider simple geometry (flat, quadratic, etc.) terrain and constant motion.

2.2 Enabling Technologies

We take a direct approach to ATAS development to avoid flow-field estimation, and its associated estimation errors and computational load. The approach, however, requires the design of a new estimation architecture and the development of new solution technologies since the

earlier flow-field based estimation architecture and its associated solution technologies cannot be applied without first computing the flow-field.

In section 2.2.1, we present a Kalman filter based direct approach to egomotion and terrain shape estimation. In sections 2.2.2 and 2.2.3, we briefly describe two key technologies in support of the approach: Singular Value Decomposition (SVD) based Least Squares (LS) algorithms and Square Root Kalman filter algorithms.

2.2.1 Kalman Filter Based Direct Approach to Egomotion and Terrain Shape Estimation

Figure 2.2-1 illustrates the Kalman filter based direct approach to egomotion and terrain shape estimation. The estimation system comprises three functional modules—an image processor, a snapshot estimator, and a Kalman filter, to generate real-time motion and terrain shape estimates.

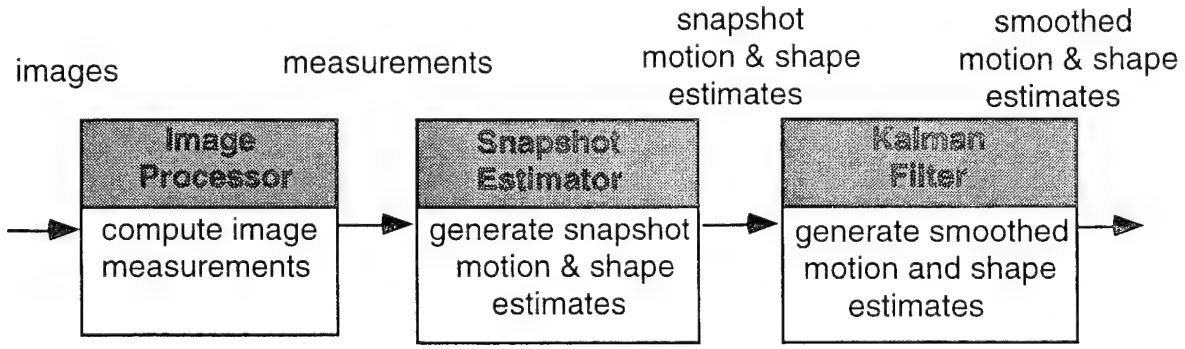


Figure 2.2-1: Egomotion and Terrain Shape Estimation System

The first stage image processor computes image measurements each time a new image is acquired. These image measurements consist of the first order image gradients and results from arithmetic operations between the image gradients and image pixel coordinates in an observer-fixed coordinate system. Consequently, except for the image gradient computation, this image processor is significantly different from a conventional, first stage flow-field computer (Negahdaripour & Lee, 1992). The flow-field computation involves complicated numerical techniques, cannot be done accurately, and is very computationally intensive. The image measurement computation, on the other hand, involves only few arithmetic operations between image gradients and precisely known pixel coordinates, and thus can be done accurately and requires much less computation than does the flow-field computation.

Using solely the current image measurements, the snapshot estimator then generates a "snapshot" (instantaneous) estimate of egomotion states and terrain shape. The accuracy and reliability that can be achieved by the snapshot estimator, however, are bounded by two factors:

1) the high signal-to-noise ratios due to violation of first-derivative continuous assumptions in real images and the limitations of digital imaging technology; and 2) the small shift between consecutive image frames that may simply not contain enough information for reliable motion and shape estimation.

The last stage a Kalman filter improves the accuracy of the snapshot estimates by integrating information over time and exploiting the continuity constraints in the observer's motion states and the terrain shape.

This modular approach provides both performance improvements and simplifications for system implementation. First, it avoids complicated interactions between low-level image processing and high-level motion state and terrain shape estimation, and consequently avoids nonlinear relationships induced by such interactions in an integrated approach. Second, separation of image processing, snapshot estimation, and Kalman filtering makes each problem simpler and its solution easier and more robust. Third, the system has a natural interface to incorporate external information for performance enhancement. For example, if the external information on the observer motion state is available, it can then be easily incorporated into the Kalman filter without reconfiguration of the entire system. Finally, we believe that this modular architecture can serve as the basis of a model of the human flow-field based vision process.

The development of the Kalman filter based direct approach calls for accurate, robust, and efficient static and dynamic estimation algorithms. The Singular Value Decomposition (SVD) Least Squares (LS) algorithm and the Square-Root Kalman filter algorithm are the two state-of-art techniques for static and dynamic estimation, and are briefly described below in sections 2.2.2 and 2.2.3.

2.2.2 Singular Value Decomposition (SVD) Least Squares (LS) Estimation Algorithm

LS estimation is one of the most widely used static estimation techniques. Modeling the noisy measurements of an unknown system (process) via a parametric model, LS estimation provides a *best* estimate of system parameters in the least squares sense. Formally speaking, let $\mathbf{z}$ represent a column vector of M unknown parameters. Given N noisy measurements of vector $\mathbf{a}_i$ and scalar b_i, and assuming a linear parametric relationship between the parameters and measurements, defined by

$$b_i = \mathbf{a}_i^T\mathbf{z} + \zeta_i,\ i = 1, 2, \cdots, N \tag{2.2-1}$$

where ζ_i represents an unknown additive noise, LS estimation determines an optimal estimate of $\mathbf{z}$, $\hat{\mathbf{z}}$ that minimizes an LS residual function $r(\mathbf{z})$, defined by

$$r(\mathbf{z}) = \sum_{i=1}^{N} [b_i - \mathbf{a}_i^T \mathbf{z}]^2 \tag{2.2-2}$$

Grouping N measurements $\{\mathbf{a}_i\}$ and $\{b_i\}$ via the single matrix and vector expressions

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 \\ \vdots \\ \mathbf{a}_N \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix} \tag{2.2-3}$$

the linear LS estimation can then be written as determining the estimate z that minimizes

$$r(\mathbf{z}) \equiv |\mathbf{b} - \mathbf{A}\mathbf{z}| \tag{2.2-4}$$

The SVD LS algorithm is the method of choice for solving most linear LS estimation problems since it provides numerically stable solution method that is not susceptible to computer roundoff error. It also provides an error covariance estimate of the LS solution itself (Press, Teukolsky, Vetterling & Flannery, 1992).

The SVD method is based on the following theorem of linear algebra: any N × M matrix, whose number of rows N is greater than or equal to its number of columns M, can be written as the product of an N × M column-orthogonal matrix **U**, an M × M diagonal matrix **W** with positive or zero elements (the singular values), and the transpose of an M × M orthogonal matrix **V**:

$$\mathbf{A} = \mathbf{U}\mathbf{W}\mathbf{V}^T \tag{2.2-5a}$$

where

$$\mathbf{U}^T\mathbf{U} = \mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}, \text{ and } \mathbf{W} = \text{diag}\,(w_1, \cdots, w_M) \tag{2.2-5b}$$

As a basic linear algebra operation, the SVD method has been implemented in various computer languages, e.g., a C computer code is given in Press et al. (1992).

Once the SVD of the **A** matrix is obtained, the LS solution $\hat{\mathbf{z}}$ of problem (2.2-4) can then be found as

$$\hat{\mathbf{z}} = \mathbf{V}\mathbf{W}^{-1}\mathbf{U}^T\mathbf{b} \tag{2.2-6}$$

The covariance **P** in the estimate of $\hat{\mathbf{z}}$ is given by

$$\mathbf{P} = \text{Cov}(\hat{\mathbf{z}}) = \mathbf{V}\mathbf{W}^{-2}\mathbf{V}^T \tag{2.2-7}$$

The SVD LS algorithm can also be extended to solve the constrained linear LS estimation problem, i.e., to determine the estimate that $\hat{\mathbf{z}}$ minimizes

$$r(\mathbf{z}) \equiv |\mathbf{b} - \mathbf{A}\mathbf{z}| \tag{2.2-8a}$$

subject to the constraint that

$$\mathbf{z}^T\mathbf{z} = 1 \tag{2.2-8b}$$

Specifically, denoting

$$\mathbf{y} \equiv \mathbf{V}^T\mathbf{z} \quad \text{and} \quad \mathbf{c} \equiv \mathbf{U}^T\mathbf{b} \tag{2.2-9a}$$

we have the LS problem (2.2-8) being transformed to determine the $\hat{\mathbf{y}}$ that minimizes

$$r(\mathbf{y}) \equiv |\mathbf{b} - \mathbf{W}\mathbf{y}| \tag{2.2-9b}$$

subject to the constraint that

$$\mathbf{y}^T\mathbf{y} = 1 \tag{2.2-9c}$$

Defining the Lagrange function

$$r(\mathbf{y}, \lambda) = |\mathbf{c} - \mathbf{W}\mathbf{y}| + \lambda(\mathbf{y}^T\mathbf{y} - 1) \tag{2.2-10}$$

and noting that $\mathbf{W}$ is a diagonal matrix, we have the parametric solution $\mathbf{y}(\lambda)$ to (2.2-10) given by

$$y_i = \frac{w_i c_i}{w_i^2 + \lambda}, \text{ for } i = 1, 2, ..., M \tag{2.2-11}$$

Substituting (2.2-11) into $\mathbf{y}^T\mathbf{y} = 1$, the Lagrange multiplier λ, is then constrained by the following single variable equation:

$$\sum_{i=1}^{M} \left(\frac{w_i c_i}{w_i^2 + \lambda}\right)^2 = 1 \tag{2.2-12}$$

The equation has a unique solution and can be found using standard root-finding algorithms, such as Newton' s method given in Press, Teukolsky, Vetterling & Flannery (1992). After the Lagrange multiplier λ is found, the original parameter vector $\hat{\mathbf{z}}$ and its estimated error covariance can be found via

$$\hat{\mathbf{z}} = \mathbf{V}\hat{\mathbf{y}} \text{ where } \hat{\mathbf{y}} = \left(\frac{w_1 c_1}{w_1^2 + \lambda}, \cdots, \frac{w_M c_M}{w_M^2 + \lambda}\right)^T \tag{2.2-13a}$$

$$\text{Cov}(\hat{\mathbf{z}}) = \mathbf{V}\ \text{diag}\left(\frac{1}{w_1^2 + \lambda}, \cdots, \frac{1}{w_M^2 + \lambda}\right)\mathbf{V}^T \tag{2.2-13b}$$

The SVD LS estimation algorithm has a computational complexity of $O(NM^2)$ flops. A flop roughly constitutes the effort of doing a floating point add, a floating point multiply, and some indexing.

2.2.3 Square-Root Kalman Filter Algorithms

The Kalman filtering approach is a Bayesian estimation technique used to obtain accurate and robust estimates of stochastic dynamic system states that are observed with noisy measurements. The implementation of a Kalman filter requires specification of three probabilistic models: system, measurement, and prior estimate (Anderson & Moore, 1979).

The *system model* describes the system state evolution over time, and is defined by the state-space transition equation

$$\mathbf{x}_{k+1} = \mathbf{A}_k\mathbf{x}_k + \boldsymbol{\varepsilon}_k, \qquad \boldsymbol{\varepsilon}_k \sim N(0, Q_k) \tag{2.2-14}$$

where $\mathbf{x}_k$ is the current system state vector at time k, $\mathbf{A}_k$ is a known transition matrix, and $\boldsymbol{\varepsilon}_k$ is a Gaussian noise with zero mean and a covariance $\mathbf{Q}_k$.

The *measurement model* relates the state vector $\mathbf{x}_k$ to the noisy measurement vector $\mathbf{y}_k$ via a measurement matrix $\mathbf{C}_k$ and the addition of a Gaussian noise $\boldsymbol{\eta}_k$ with zero mean and a covariance $\mathbf{R}_k$. It is defined by the measurement equation

$$\mathbf{y}_k = \mathbf{C}_k\mathbf{x}_k + \boldsymbol{\eta}_k, \qquad \boldsymbol{\eta}_k \sim N(0, \mathbf{R}_k) \tag{2.2-15}$$

The prior estimate model describes the knowledge about the initial system state $\hat{\mathbf{x}}_0$ and its covariance $\mathbf{P}_0$ before the first measurement is taken, and assumes uncorrelated system and measurement noises. It is defined by

$$\hat{\mathbf{x}}_0 = E[\mathbf{x}_0] \tag{2.2-16a}$$

$$\mathbf{P}_0 = CovE[\mathbf{x}_0] \quad \text{and} \quad E[\boldsymbol{\varepsilon}_k\boldsymbol{\eta}_k^T] = 0 \tag{2.2-16b \& c}$$

We always assume that there is no initial knowledge on the system states before the first estimate. In other words, the prior model for the system states is fixed, and is always given by

$$\hat{\mathbf{x}}_0 = 0, \mathbf{P}_0 = \text{diag}(\infty) \tag{2.2-16d}$$

Once the system, measurement, and prior models are specified, the Kalman filter algorithm operates in two phases, prediction and correction, as shown in figure 2.2-2 via the right and left blocks separated by a dashed line.

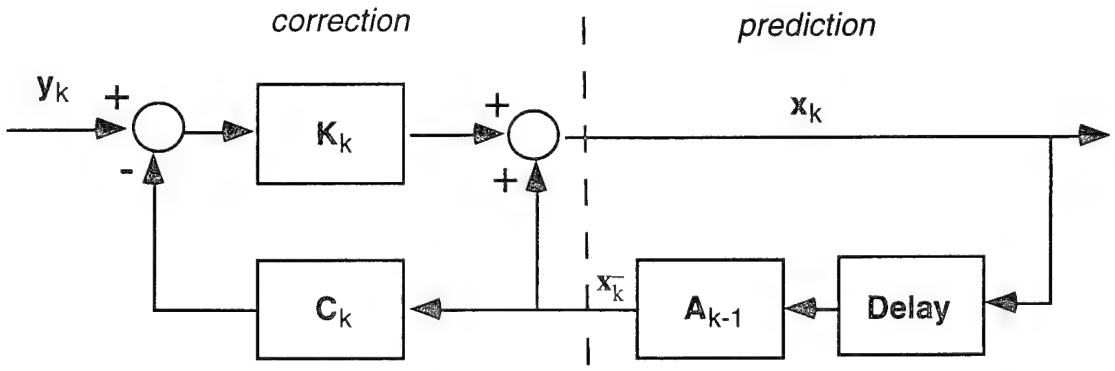


Figure 2.2-2: Kalman Filter Block Diagram

In the prediction phase, the previous state estimate $\hat{\mathbf{x}}_{k-1}$ and covariance estimate $\mathbf{P}_{k-1}$ are extrapolated to predict the current state $\mathbf{x}_k^-$ and $\mathbf{P}_k^-$ via state extrapolation and covariance extrapolation defined by

$$\hat{\mathbf{x}}_k^- = \mathbf{A}_{k-1}\hat{\mathbf{x}}_{k-1} \tag{2.2-17}$$

$$\mathbf{P}_k^- = \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1} \tag{2.2-18}$$

In the correction phase, the predicted covariance is used to compute the Kalman gain matrix $\mathbf{K}_k$ and to update the covariance matrix $\mathbf{P}_k$ via

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{C}_k^T[\mathbf{C}_k\mathbf{P}_k^-\mathbf{C}_k^T + \mathbf{R}_k]^{-1} \tag{2.2-19}$$

$$\mathbf{P}_k = [\mathbf{I} - \mathbf{K}_k^-\mathbf{C}_k]\mathbf{P}_k^- \tag{2.2-20}$$

The measurement residual $\mathbf{y}_k - \mathbf{C}_k\hat{\mathbf{x}}_k^-$ is then weighted by the gain matrix $\mathbf{K}_k$ and added to the predicted state $\mathbf{x}_k^-$ to yield the updated state estimate $\hat{\mathbf{x}}_k$ via

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k[\mathbf{y}_k - \mathbf{C}_k\hat{\mathbf{x}}_k^-] \tag{2.2-21}$$

The conventional Kalman filter algorithm is defined by equations (2.2-17) to (2.2-21). However, it is prone to serious numerical difficulties that can cause a loss of positive definiteness in updating the covariance matrix $\mathbf{P}_k$ leading to divergent state estimates. Although algebraically equivalent to the conventional Kalman filter algorithm, square-root Kalman filter algorithms exhibit improved numerical precision and stability, especially in those ill-conditioned problems where conventional algorithms fail.

Square root Kalman filter algorithms are developed using a square root decomposition feature of symmetric, positive semidefinite matrices—the class of matrices which the covariance matrix $\mathbf{P}_k$ belongs (Maybeck, 1979). This feature dictates that for each $\mathbf{P}_k$ there exists at least one

square root matrix $\sqrt{\mathbf{P}_k}$ such that

$$\mathbf{P}_k = \sqrt{\mathbf{P}_k}\sqrt{\mathbf{P}_k}^T \tag{2.2-22}$$

The basic idea of the square root algorithms is to replace the recursive updates of $\mathbf{P}_k$ in both the prediction and correction phases with the updates of $\sqrt{\mathbf{P}_k}$, and to update the state estimate using an optimal gain computed using $\sqrt{\mathbf{P}_k}$ instead of $\mathbf{P}_k$. The update of $\sqrt{\mathbf{P}_k}$ guarantees a positive definite covariance matrix $\mathbf{P}_k$, thus eliminating the possibility of state estimate divergence. Various versions of square root Kalman filter algorithm implementation are given in Maybeck (1979).

3. PASSIVE SENSOR MOTION AND TERRAIN AWARENESS SYSTEM (ATAS)

This chapter provides a description of the passive sensor motion and terrain awareness system (ATAS). Section 3.1 presents the architecture of the ATAS that is composed of an image sensor and three processors: an image processor, a snapshot estimator, and a Kalman filter. Section 3.2 describes the basis for the flow-based egomotion and terrain shape estimation problem, and derives the basic constraint relating unknown motion and terrain shape parameters to directly measurable image gradients. Section 3.3 then describes the image processor, section 3.3 the snapshot estimator, and section 3.4 the Kalman filter.

3.1 System Architecture

Figure 3.1-1 shows the architecture of the passive sensor altitude and terrain awareness system. The architecture is developed using a Kalman filter based direct approach, and is composed of an image sensor and three processors: an image processor, a snapshot estimator, and a Kalman filter, as shown in figure 3.1-1.

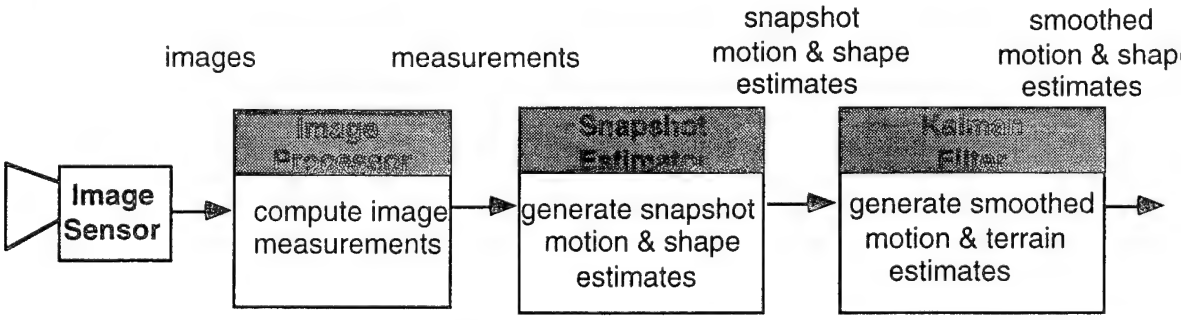


Figure 3.1-1: Passive Sensor Altitude and Terrain Awareness System

The **image sensor** provides sequences of real-time two dimensional dynamic images of the overflown terrain. We simulated real-time imaging operations using CGI and pre-recorded imagery. With CGI, sequential image frames were either fed directly to the image processor or were sent to a video recorder for storage and later use. With the pre-recorded imagery, a real-time frame grabber operated on the video signal to produce the necessary sequence of image frames, which were then fed direcetly to the image processor. The simulation of the image sensor/digitizer is described in greater detail later in section 4.1.

The **image processor** computes image measurements each time a new image (frame) is acquired. These image measurements consist of first order image gradients and results from arithmetic operations between the image gradients and the image pixel coordinates. Based soley on the current image measurements, the **snapshot estimator** then generates a "snapshot"

estimate of motion states and terrain shape. Using the snapshot estimates as state measurements, the **Kalman filter** then improves estimation accuracy by integrating information over time and exploiting the continuity constraints in the observer's motion states and the terrain shape.

3.2 Problem Formulation and Estimation Constraint Equation

Consider an observer undergoing both translational and rotational motion over motionless terrain as illustrated in figure 3.2-1. The observer's linear and angular velocity are denoted by the vectors **V** and Ω, respectively. Let P_i be a viewed point on the terrain, located at a range ρ_i from the observer. In the observer's frame of reference, the unit-length line-of-sight (LOS) direction vector $\mathbf{u}_i = \boldsymbol{\rho}_i/|\boldsymbol{\rho}_i|$ will appear to change with time, at a rate given by

$$\dot{\mathbf{u}}_i = \boldsymbol{\omega}_i \times \mathbf{u}_i \tag{3.2-1a}$$

where $\boldsymbol{\omega}_i$ is the rotation rate of the LOS vector (effectively, the 3D visual flow seen by the observer), and is shown by Zacharias and Levison (1980) to be given as

$$\boldsymbol{\omega}_i = \mathbf{u}_i \times (\mathbf{u}_i \times \boldsymbol{\Omega}) - \frac{1}{\tau_i}(\mathbf{u}_i \times \mathbf{u}_v) \tag{3.2-1b}$$

where $\mathbf{u}_v$ is the observer's *heading vector*, and τ_i is the *impact time*, defined by

$$\mathbf{u}_v = \frac{\mathbf{V}}{|\mathbf{V}|}, \qquad \tau_i = \frac{\rho_i}{|\mathbf{V}|} \tag{3.2-2a,b}$$

The unit length heading vector defines only the direction of the observer's motion, but not his speed. The impact time is the elapsed time before the observer impacts with the surface at point P_i, *if* the observer were to head directly at P_i, at the speed $|\mathbf{V}|$.

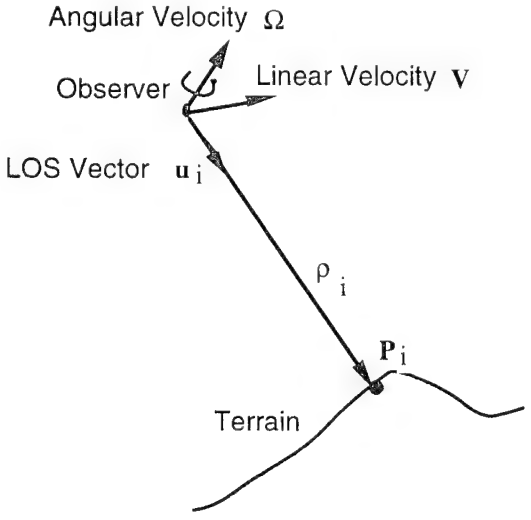


Figure 3.2-1: Line-of-Sight Rate

We now develop a model for estimating egomotion and terrain shape based on the image gradient information. The model employs two additional key relationships that relate the directly observable image intensities captured on a hypothesized image plane, to the unknown motion/terrain variables.

The first key relationship relates the 3D visual flow to its 2D projection on a hypothesized image plane. Consider the 2D image of the terrain formed on an image plane within the observer's field-of-view (FOV), as illustrated in figure 3.2-2. Let the focal point be the origin of a screen coordinate system. Let $\mathbf{u}_s$ denote the unit length vector normal to the sensor plane, and d_s denote the effective focal length. The image of point P_i, given by P_i', located by the vector $\mathbf{d}_i$ in the image plane, is defined by the focal-length normalized x-y coordinates $\boldsymbol{\delta}_i = \mathbf{d}_i/d_s = (x_i, y_i, 1)^T$. As shown by Zacharias, Miao & Riley (1992), the rotation rate of the LOS vector $\mathbf{u}_i$ generates the following rate of change of this in-plane location vector $\boldsymbol{\delta}_i$:

$$\dot{\boldsymbol{\delta}}_i = \frac{\boldsymbol{\omega}_i \times \mathbf{u}_s}{(\mathbf{u}_i \cdot \mathbf{u}_s)^2} = \mathbf{H}_{i\Omega}\boldsymbol{\Omega} + \mathbf{H}_{iv}\frac{\mathbf{u}_v}{\tau_{iz}} \tag{3.2-3a}$$

where $\tau_{iz} = \tau_i / |\boldsymbol{\delta}_i| = \tau_i / \sqrt{1 + x_i^2 + y_i^2}$ is the plane normal (screen coordinate system) impact time, $\boldsymbol{\delta}_i$ is a two dimensional vector in the screen x-y plane, and

$$\mathbf{H}_{i\Omega} = \begin{pmatrix} x_i y_i & -(x_i^2 + 1) & y_i \\ (y_i^2 + 1) & -x_i y_i & -x_i \end{pmatrix} \tag{3.2-3b}$$

$$\mathbf{H}_{iv} = \begin{pmatrix} -1 & 0 & x_i \\ 0 & -1 & y_i \end{pmatrix} \tag{3.2-3c}$$

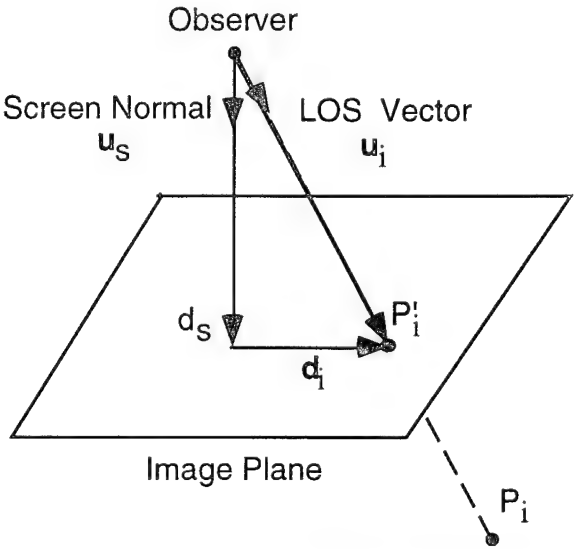


Figure 3.2-2: Image Plane Geometry

The second key relationship relates 3D visual flow to the instantaneous spatial and temporal image gradients in the 3D image plane. Consider how the intensity I_i of image point P_i' changes with time at an in-plane location $\boldsymbol{\delta}_i$. Using the partial derivative rule, we obtain the following expression for the total temporal derivative of image intensity:

$$\frac{dI(x_i, y_i, t)}{dt} = I_{ix}\frac{dx_i}{dt} + I_{iy}\frac{dy_i}{dt} + I_{it} \tag{3.2-4}$$

where $I_{ix} = \partial I_i/\partial x$ and $I_{iy} = \partial I_i/\partial y$, and we have assumed the image brightness function is first order continuous at the image point $\boldsymbol{\delta}_i$. Assuming that the brightness of any terrain point is constant over time, we require the total derivative of (3.2-4) to be zero, so that

$$I_{ix}\frac{dx_i}{dt} + I_{iy}\frac{dy_i}{dt} + I_{it} = 0 \text{ or } I_{it} = -(I_{ix}, I_{iy})\boldsymbol{\delta}_i \tag{3.2-5}$$

where $\boldsymbol{\delta}_i$ is given by eq. (3.2-3a). Substituting (3.2-3a) into (3.2-5), for each pixel i, we obtain an *estimation constraint equation* that relates the directly measurable image intensity I_i (through its gradients I_{it}, I_{ix}, I_{iy}), with the observer motion states $\boldsymbol{\Omega}$, $\mathbf{u}_v$, and terrain impact time τ_i as follows:

$$I_{it} = \mathbf{h}_{i\Omega}^T\boldsymbol{\Omega} + \frac{1}{\tau_{iz}}\mathbf{h}_{iv}^T\mathbf{u}_v \tag{3.2-6a}$$

where

$$\mathbf{h}_{i\Omega} = \mathbf{H}_{i\Omega}\begin{pmatrix} I_{ix} \\ I_{iy} \end{pmatrix} \text{ and } \mathbf{h}_{iv} = \mathbf{H}_{iv}\begin{pmatrix} I_{ix} \\ I_{iy} \end{pmatrix} \tag{3.2-6b}$$

3.3 Image Processor

The image processor computes the image measurements $\{\mathbf{h}_{i\Omega}, \mathbf{h}_{iv}, I_{it}\}$ each time a new image is acquired. The processor boils down to computing the image gradients, since computing $\{\mathbf{h}_{i\Omega}, \mathbf{h}_{iv}\}$ via (3.2-3b) involves only arithmetic operations between image gradients and image pixel coordinates. We use a direct numerical differentiation method to compute the image gradients. Specifically, Let k be the current time (image frame) index, and Δt be the frame interval. At the current time k, we have image k defined by its intensity map $\{I(x_i, y_i, t)\}$, where index i denotes the ith pixel in the image. The image gradients are computed for each pixel using the following two-point central difference formulas:

$$I_{ix} = \frac{I(x_i+\Delta x, y_i, t - \Delta t) - I(x_i - \Delta x, y_i, t - \Delta t)}{2\Delta x}$$

$$I_{iy} = \frac{I(x_i, y_i + \Delta y, t - \Delta t) - I(x_i, y_i - \Delta y, t - \Delta t)}{2\Delta y}$$

$$I_{it} = \frac{I(x_i, y_i, t) - I(x_i, y_i, t - 2\Delta t)}{2\Delta t} \tag{3.3-1}$$

where Δt is the temporal sampling time, and Δx and Δy are the x and y pixel dimensions. The two-point central formula provides higher accuracy than the standard two-point backward formula ($O(\Delta x)^2$ versus $O(\Delta x)$), at the cost of storing one additional old image frame. The cost of this is relatively low. Moreover, since the temporal and spatial gradients are estimated at the same (delayed) point in time, no phase shift is introduced. After I_{ix} and I_{iy} are computed, the measurements $\mathbf{h}_{i\Omega}$ and $\mathbf{h}_{iv}$ are computed using (3.2-3b).

Three major processes contribute to image measurement (gradient) errors: the spatial, temporal, and intensity quantizations performed in generating the digitized image frames and in computing the gradients. Assuming that the errors induced by the three types of quantization are independent, it can be shown that the average error $\bar{\Delta}$ and variance σ^2 in the measurements are as follows:

$$\bar{\Delta}(I_{ix}) = -\frac{\Delta_i^2}{6} I'''_{x3}(x_i, y_i)$$

$$\bar{\Delta}(I_{iy}) = -\frac{\Delta_i^2}{6} I'''_{y3}(x_i, y_i)$$

$$\sigma^2(I_{ix}) = \frac{\Delta_i^6 [I''''_{x4}(x_i, y_i)]^2}{108} + \frac{1}{24G^2\Delta_i^2}$$

$$\sigma^2(I_{iy}) = \frac{\Delta_i^6 [I''''_{y4}(x_i, y_i)]^2}{108} + \frac{1}{24G^2\Delta_i^2}$$

$$\bar{\Delta}(I_{it}) = \frac{(\Delta t)^2}{6} \sum_{n=0}^{3} C_3^n I'''_{x^{3-n}y^n}(x_i, y_i) \dot{x}_i^{3-n} \dot{y}_i^n$$

$$\sigma^2(I_{it}) = \frac{(\Delta t)^6 [\sum_{n=0}^{4} C_4^n I''''_{x^{4-n}y^n}(x_i, y_i) \dot{x}_i^{4-n} \dot{y}_i^n]^2}{108} + \frac{1}{24G^2(\Delta t)^2} \tag{3.3-2}$$

where $\Delta_i = \Delta_{ix} = \Delta_{iy}$ is the pixel size, C_3^n is the binomial-coefficient, and G is the image gray level. The derivation of the formulas are provided in Appendix A.

The average error and variance formulas (3.3-2) indicate that the average errors and error variances are in proportion to the high-order image gradients. Consequently, if the image intensity function is discontinuous at higher order, a larger error will occur. In such a situation, a low-pass filter is needed to smooth the images before the gradient can be computed using the standard differentiation formula.

The application of a low-pass filter should not be abused, however, since the low-pass filtering also reduces the absolute value of image gradients. Consequently, the image signal-to-noise ratio may be reduced instead of enhanced through low-pass filtering. In addition, low-pass filtering adds computational expense to image measurement computation.

The complete image processor algorithm is given as follows:

Algorithm: Two Point Center Difference

For i = 1, ..., N

Step 1: Compute image gradients I_{ix}, I_{iy}, and I_{it} from (3.3-1)

Step 2: Compute the image measurements $\mathbf{h}_{i\Omega}$ and $\mathbf{h}_{iv}$ from (3.2-3b)

Computation of image measurements using this algorithm requires 13N flops, where 3N flops are for gradient computation, and 10N flops are for image measurement computation. A flop roughly constitutes the effort of doing a floating point add, a floating point multiply, and some indexing.

Image measurement computation can be done in parallel since computation for each pixel can be performed independently. Consequently, if a P processor parallel computer is available for the computation, a computation reduction of P times can then be achieved.

3.4 Snapshot Estimator

The objective of the snapshot estimator is to determine, each time an image is acquired, the snapshot (instantaneous) motion and terrain estimates $\hat{\boldsymbol{\Omega}}$, $\hat{\mathbf{u}}_v$,and $\{\hat{\tau}_{iz}\}$ which minimize

$$r(\boldsymbol{\Omega}, \mathbf{u}_v, \{\tau_{iz}\}) \equiv \sum_{i=1}^{N} \left| I_{it} - (\mathbf{h}_{i\Omega}^T \boldsymbol{\Omega} + \frac{1}{\tau_{iz}} \mathbf{h}_{iv}^T \mathbf{u}_v) \right|^2 \tag{3.4-1}$$

given N measurements $\{\mathbf{h}_{i\Omega}, \mathbf{h}_{iv}, I_{it}\}$. Direct inspection of (3.4-1), however, reveals that satisfaction of (3.4-1) alone cannot provide a unique solution to the snapshot estimation problem since there are N + 5 unknown motion and terrain shape variables to be found from N constraints. We resolve this "under-constraint" difficulty by requiring that the terrain-impact time map $\{\tau_{iz}\}$ satisfy additional terrain structural constraints.

This can be done if we impose on the terrain impact time map $\{\tau_{ia}\}$ a parametric terrain structure constraint specified by M structural parameters, where M < N-5. Subject to this additional structural constraint, the snapshot estimation problem then becomes one of solving (M+5) motion and terrain shape variables from N overconstrained equations, where the excess constraint equations are used to improve the accuracy of the solution.

Two types of parametric terrain structure constraints are considered. The first type explicitly requires that the terrain impact time map $\{\tau_{iz}\}$ satisfy a polynomial geometry model such as flat surface, quadratic surface, etc. The second type does not assume a priori a surface surface model, but requires that the terrain impact time map $\{\tau_{iz}\}$ satisfy a pre-determined impact-time (depth) layer structure. Section 3.4.1 describes the geometry based snapshot estimator, while section 3.4.2 describes the structure based snapshot estimator.

3.4.1 Geometry Based Snapshot Estimator

Figure 3.4-1 illustrates the (polynomial) geometry-based approach to snapshot estimation. Assuming that the terrain surface can be modeled by a polynomial geometry function, the approach determines an LS solution to the model parameters and motion state parameters.

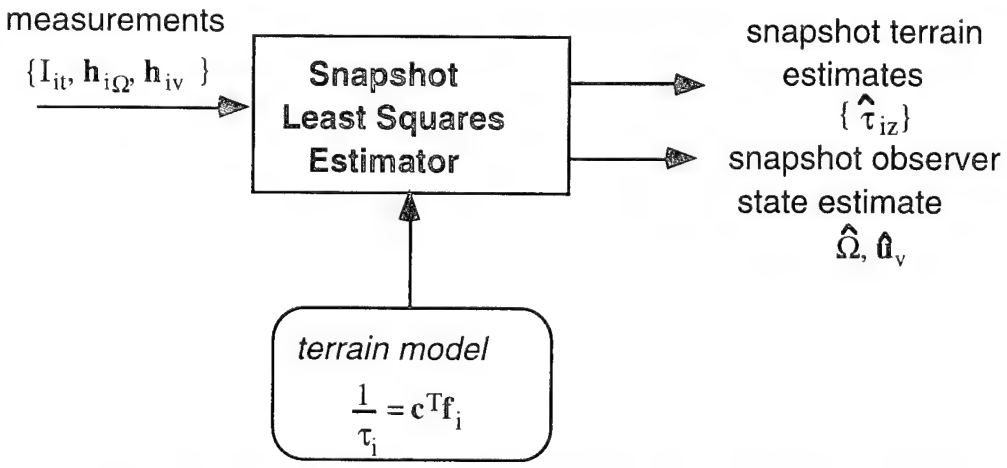


Figure 3.4-1: Geometry Baed Approach to Snapshot Estimation

We begin by denoting a terrain point $\mathbf{P}_i$ by its impact time vector

$$\boldsymbol{\tau}_i \equiv \frac{\mathbf{P}_i}{|\mathbf{V}|} = (\tau_{ix}, \tau_{iy}, \tau_{iz})^T \tag{3.4-2}$$

relative to an observer-fixed coordinate system. Under perspective projection, the surface point $\mathbf{P}_i$ projects to a point in the normalized image plane

$$(x_i, y_i)^T = (\tau_{ix}/\tau_{iz}, \tau_{iy}/\tau_{iz})^T \tag{3.4-3}$$

We model the inverse of the z-axis terrain impact time τ_{iz} as an n-th order polynomial of the in-plane image coordinates (x_i, y_i), given by

$$\frac{1}{\tau_{iz}} = \mathbf{c}^T\mathbf{f}_i \tag{3.4-4a}$$

where

$$\mathbf{c} = (c_0, c_2, \cdots, c_{n^2+3n/2})^T \tag{3.4-4b}$$

$$\mathbf{f}_i = (1, x_i, y_i, \cdots, x_i^n, x_i^{n-1}u_i, \cdots, y_i^n)^T \tag{3.4-4c}$$

where $\mathbf{c}$ is the model parameter vector of dimension $M = n^3 + 3n/3$, and $\mathbf{f}_i$ is the image coordinate polynomial function. We choose the inverse impact time model (3.4-4a) because the impact time appears in the estimation constraint by its inverse. Transformation of the terrain model from the inverse impact time format to normal impact time format, and from image coordinate representation (x_i, y_i) to the terrain coordinate representation (τ_{ix}, τ_{iy}) can be found in Negahdaripour & Lee, (1992).

Requiring that the terrain points (impact times) satisfy the geometry model (3.4-4a) means that (3.4-4a) can be substituted into (3.4-1). After substitution, the simplified snapshot estimation problem becomes one of determining the $\hat{\mathbf{c}}$, $\hat{\boldsymbol{\Omega}}$, and $\hat{\mathbf{u}}_v$ which minimize:

$$r(\mathbf{c}, \boldsymbol{\Omega}, \mathbf{u}_v) \equiv \sum_{i=1}^{N} \left| I_{it} - \mathbf{h}_{i\Omega}^T \boldsymbol{\Omega} - \mathbf{c}^T \mathbf{f}_i \mathbf{h}_{iv}^T \mathbf{u}_v \right|^2 \tag{3.4-5}$$

across all *feasible* model parameters $\mathbf{c}$, the angular velocity $\boldsymbol{\Omega}$, and heading vector $\mathbf{u}_v$.

The residual function $r(\mathbf{c}, \boldsymbol{\Omega}, \mathbf{u}_v)$ of (3.4-5) is a bilinear function of the model parameter vector $\mathbf{c}$, angular velocity $\boldsymbol{\Omega}$, and heading vector $\mathbf{u}_v$. In other words, if *either* $(\mathbf{c}, \Omega)$ or $\mathbf{u}_v$ were known, estimation of $\mathbf{u}_v$ or $(\mathbf{c}, \Omega)$ would be an easily solvable linear LS minimization problem. We employ a two stage integration approach to estimate $(\mathbf{c}, \Omega)$ and $\mathbf{u}_v$.

Specifically, given an estimate of the heading vector $\hat{\mathbf{u}}_v$, the snapshot terrain shape estimation problem of (3.4-1) becomes a linear LS estimation problem to determine $\hat{\boldsymbol{\Omega}}$ and $\hat{\mathbf{c}}$ which minimize

$$r(\boldsymbol{\Omega}, \mathbf{c}) = \left| \mathbf{I}_t - \begin{bmatrix} \vdots & \vdots \\ \mathbf{h}_{i\Omega}^T & (\hat{\mathbf{u}}_v^T \mathbf{h}_{iv}) \mathbf{f}_i^T \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} \boldsymbol{\Omega} \\ \mathbf{c} \end{bmatrix} \right|^2 \tag{3.4-6}$$

The snapshot estimates $\hat{\boldsymbol{\Omega}}$ and $\hat{\mathbf{c}}$ and their covariances $\mathbf{P}_\Omega$ and $\mathbf{P}_c$, can then be found using the SVD LS algorithm, presented earlier in section 2.2.

The second stage assumes angular velocity and model coefficient vector estimates $(\hat{\boldsymbol{\Omega}}, \hat{\mathbf{c}})$, so that the snapshot terrain shape estimation problem of (3.4-1) becomes a constrained linear LS estimation problem to determine

$$\hat{\mathbf{u}}_v \quad \text{which minimizes} \quad r(\mathbf{u}_v) \equiv \left|\mathbf{b}_t - \begin{bmatrix} \vdots \\ (\hat{\mathbf{c}}^T\mathbf{f}_i)\mathbf{h}_{iv}^T \\ \vdots \end{bmatrix} \mathbf{u}_v\right|^2 \text{ subject to } \mathbf{u}_v^T\mathbf{u}_v = 1 \tag{3.4-7a}$$

where

$$\mathbf{b}_t \equiv \mathbf{I}_t - \mathbf{H}_\Omega\hat{\Omega} \tag{3.4-7b}$$

The snapshot estimate $\hat{\mathbf{u}}_v$ and its covariance $\mathbb{P}_v$ can then be found using constrained SVD LS algorithm, presented earlier in section 2.2.

Figure 3.4-2 summarizes the block diagram of this two stage integration approach. Starting with an initial guess on $\mathbf{u}_v$, the snapshot terrain and angular velocity estimator generates an estimate of the terrain model parameters $\mathbf{c}$ and Ω, which in turn drive the snapshot heading estimator to update the estimate on $\mathbf{u}_v$. The updated heading estimate $\mathbf{u}_v$ is then fed back to the snapshot terrain and angular velocity estimator to generate a new estimate of the terrain model parameters and the angular velocity. The iteration proceeds until a convergence criterion on residual function reduction is satisfied. After the convergence, the impact time map estimate $\hat{\tau}_{iz}$ and its variance $\sigma(\hat{\tau}_{iz})$ can then be determined via

$$\hat{\tau}_{iz} = \frac{1}{\hat{\mathbf{c}}^T\mathbf{f}_i}$$

$$\sigma(\hat{\tau}_{iz}) = \frac{\mathbf{f}_i^T\mathbf{P}_c\mathbf{f}_i}{\hat{\tau}_{iz}^2} \tag{3.4-8a, b}$$

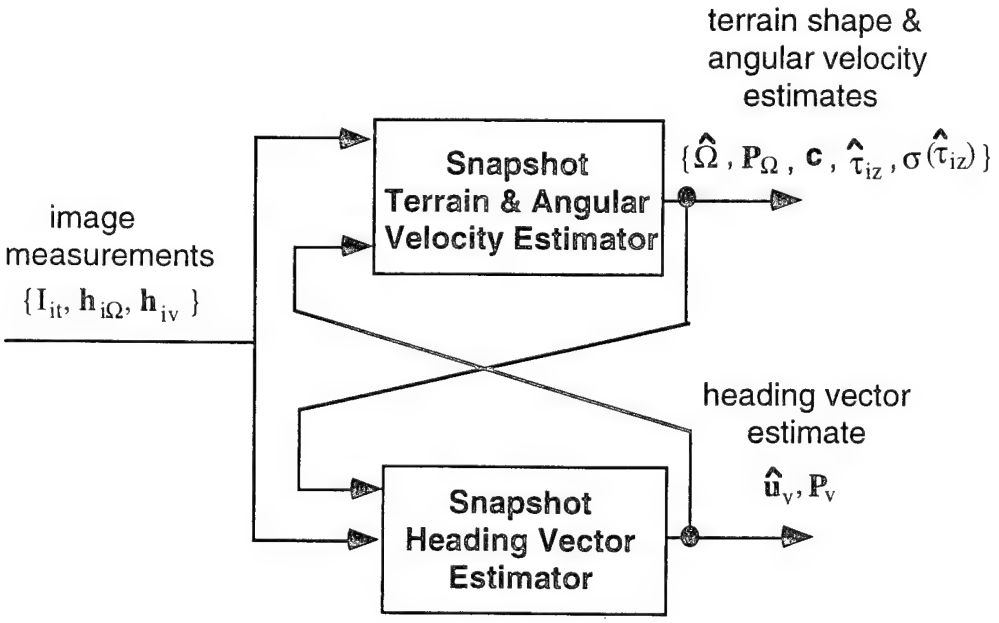


Figure 3.4-2: Geometry-Based Snapshot Estimator

Note that a pure guess on $\mathbf{u}_v$ is needed only for the first snapshot estimate. The snapshot estimate at any other later time (frame) can always use the value of the previous $\mathbf{u}_v$ as a starting estimate since the observer motion states, in general, do not change much from one image frame to another.

The complete geometry model based snapshot estimation algorithm is given as follows:

Algorithm: Geometry-Based Snapshot Estimator

1) Initiate
If it is the first time the algorithm is called, select an initial (unit length) $\mathbf{u}_v$
2) Continue snapshot estimation until residue function r converges
Call SVD LS algorithm to estimate model parameters $\hat{\mathbf{c}}$ and angular velocity $\hat{\boldsymbol{\Omega}}$
Call constrained SVD LS algorithm to estimate heading vector $\hat{\mathbf{u}}_v$

Terrain and angular velocity estimation complexity is of the order $(M+3)^2N$ flops, while heading estimation complexity is of the order 9N flops. If the geometry based snapshot estimator takes L iterations to converge, then overall computational complexity is of the order $L((M+3)^2+9)N$ flops.

3.4.2 Structure-Based Snapshot Estimator

The structure-based snapshot estimator models the terrain surface as multiple depth layers, each containing terrain points of the same depth (impact time). Given the number of layers allowed to define a terrain surface, the estimator first optimally assigns terrain points among layers (or determines the optimal terrain structure) before making the motion and terrain shape estimates.

We use a structure of M equal distance layers to define a terrain, where $S = \{1\Delta, 2\Delta, ..., m\Delta, ..., M\Delta\}$ and Δ is a given constant defining the relative distance between layers. Note that the structure is defined independent of the coordinate system and in relative distance terms. Requiring that each terrain point be assigned to one of the M layers, we formulate the LS snapshot estimation problem to minimize the following residual function $r(\{a_i\}, \boldsymbol{\Omega}, \mathbf{y})$ across all feasible assignments of terrain points $\{a_i\}$, all candidate angular velocities Ω, and all candidate motion-scalar variables $\mathbf{y} \equiv \mathbf{u}_v/\tau$

$$r(\{a_i\}, \boldsymbol{\Omega}, \mathbf{y}) \equiv \left\| \begin{bmatrix} \vdots \\ I_{it} \\ \vdots \end{bmatrix} - \begin{bmatrix} \vdots & \vdots \\ \mathbf{h}_{i\Omega}^T & a_i \mathbf{h}_{iv}^T \\ \vdots & \vdots \end{bmatrix} \begin{pmatrix} \boldsymbol{\Omega} \\ \mathbf{y} \end{pmatrix} \right\|^2$$

$$= \left| \mathbf{I}_t - [\mathbf{H}_\Omega, \mathbf{H}_v] \begin{pmatrix} \boldsymbol{\Omega} \\ \mathbf{y} \end{pmatrix} \right|^2 \tag{3.4-9}$$

where the (N×6) matrix $[\mathbf{H}_\Omega, \mathbf{H}_v]$ is determined by the image measurements and the terrain point assignment policy. The only difference between (3.4-9) and the original snapshot estimation problem (3.4-1) is that in (3.4-1) the impact times can be any real number, but in (3.4-8) they are limited to a discrete set of values defined by the terrain structure S.

Notice that the angular velocity measurement matrix $\mathbf{H}_\Omega$ does not depend on the terrain point assignments. The LS solution to Ω is given by

$$\hat{\boldsymbol{\Omega}} = \mathbf{V}_\Omega \text{diag}\,(1/\lambda_1, 1/\lambda_2, 1/\lambda_3)\mathbf{U}_\Omega^T (\mathbf{I}_t - \mathbf{H}_v\mathbf{y}) \tag{3.4-10}$$

where $\mathbf{V}_\Omega$, diag $(\lambda_1, \lambda_2, \lambda_3)$, and $\mathbf{U}_\Omega$ are the SVD of $\mathbf{H}_\Omega$. Substituting (3.4-10) into (3.4-9), the problem becomes one of minimizing the following residual function $r(\{a_i\}, \mathbf{y})$ across all feasible assignments of terrain points and all candidate motion-scalar variables $\mathbf{y}$:

$$r(\{a_i\}, \mathbf{y}) \equiv \left| (\mathbf{I} - \mathbf{U}_\Omega\mathbf{U}_\Omega^T)\mathbf{I}_t - (\mathbf{I} - \mathbf{U}_\Omega\mathbf{U}_\Omega^T)\mathbf{H}_v\mathbf{y} \right|^2 \tag{3.4-11}$$

Approximating the projection matrix $\mathbf{U}_\Omega\mathbf{U}_\Omega^T$ by its diagonal components, the snapshot estimation problem (3.4-11) reduces to determining $\hat{\mathbf{y}}$ and $\{\hat{a}_i\}$ which minimize

$$r(\mathbf{y}, \{a_i\}) = \left| \begin{bmatrix} \vdots \\ b_i \\ \vdots \end{bmatrix} - \begin{bmatrix} \vdots \\ a_i\mathbf{h}_i^T \\ \vdots \end{bmatrix} \mathbf{y} \right|^2$$

$$= |\mathbf{b} - \mathbf{H}\mathbf{y}|^2 \tag{3.4-12}$$

where $b_i = \text{diag}_i(\mathbf{I} - \mathbf{U}_\Omega\mathbf{U}_\Omega^T)\mathbf{I}_{it}$ and $\mathbf{h}_i = \text{diag}_i(\mathbf{I} - \mathbf{U}_\Omega\mathbf{U}_\Omega^T)\mathbf{h}_{iv}$.

Figure 3.4-4 illustrates basic ideas to convert problem (3.4-12) into a pure structure extraction problem. Given a terrain structure, the terrain and heading estimation problem (3.4-12) can be transformed into two sub-problems of: 1) finding the optimal terrain structure, and 2) determining the optimal motion-scalar variable, where the second problem can be solved in parametric terms as a function of the first problem. Removing the motion-scalar variable via the parametric representation, we reach a pure structure extraction problem that finds the optimal terrain structure without knowing either the motion states or the terrain impact time (depth).

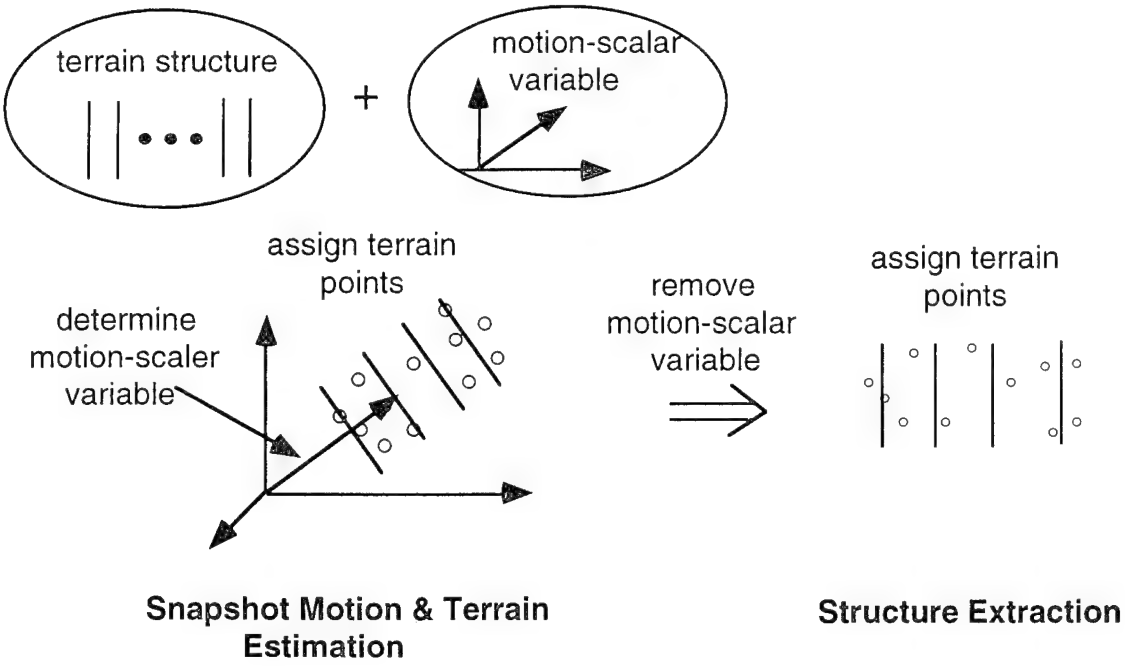


Figure 3.4-4: Problem Formulation of Structure Extraction

Specifically, for any given terrain point assignment policy $\{a_i\}$, the LS solution to the motion-scalar variable $\mathbf{y}$ is given by

$$\mathbf{y} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{b} \tag{3.4-13}$$

Substituting (3.4-13) back into (3.4-12) to remove the dependence on $\mathbf{y}$, problem (3.4-12) becomes one of assigning terrain points among the structure layers to minimize the residual

$$r(\{a_i\}) = \mathbf{b}^T\mathbf{b} - \mathbf{b}^T\mathbf{H}[\mathbf{H}^T\mathbf{H}]^{-1}\mathbf{H}^T\mathbf{b} \tag{3.4-14}$$

This problem is equivalent to maximizing the overall projection length $\mathbf{b}^T\mathbf{H}[\mathbf{H}^T\mathbf{H}]^{-1}\mathbf{H}^T\mathbf{b}$, since $\mathbf{b}^T\mathbf{b}$ is not affected by the assignment policy $\{a_i\}$. Problem (3.4-14) is thus a pure structure extraction problem since it depends neither on the heading vector nor the actual terrain impact times (depth). Its solution provides only a relative description of the terrain structure (which terrain point is at which layer), but not its actual impact time (depth). The significance of the formulation is that it breaks the chicken-and-egg circle between structure extraction and state/depth estimation by using a relative structural representation.

We now develop an iterative algorithm that optimally assigns N terrain points among M structure layers, one point at each iteration. Let n be the current iteration, $\hat{a}_i^n$ be the point i's current assignment, and $e(\{\hat{a}_i^n\})$ be the current projection length. At iteration n+1, we would change the point i's assignment to $\hat{a}_i^{n+1} = 1, 2, ..., M$. Define

$$\mu^n \equiv \mathbf{h}_i^T(\mathbb{H}^{nT}\mathbb{H}^n)^{-1}\mathbf{h}_i$$

$$\gamma_i^n = \mathbf{b}^T\mathbb{H}^n(\mathbb{H}^{nT}\mathbb{H}^n)^{-1}\mathbf{h}_i \qquad (3.4\text{-}15a, b)$$

Appendix B shows that the optimal assignment $\hat{a}_i^{n+1}$ is the solution to the following quadratic function

$$[\hat{a}_i^n\mu_i^n b_i - \gamma_i^n]\mu_i^n b_i(\hat{a}_i^{n+1} - \hat{a}_i^n)^2 + [\mu_i^n b_i^2 - (\gamma_i^n)^2](\hat{a}_i^{n+1} - \hat{a}_i^n) + [b_{it} - \hat{a}_i^n\gamma_i^n]\gamma_i^n = 0 \qquad (3.4\text{-}16)$$

when $\hat{a}_i^{n+1}$ can be any value. In other words, the optimal assignment $\hat{a}_i^{n+1}$ can be determined by comparing the projected length reduction between only two assignments with a few arithmetic operations, independent of the number of layers M which is usually large. The optimal assignment is then a layer that belongs to the structure S, and is closest to the largest projection length reduction assignment .

Note that the optimal assignment at each iteration always increases the overall projection length when the assignment is different from the previous one. On the other hand, the projection length increment is upper bounded by $\mathbf{b}_t^T\mathbf{b}_t$ and cannot increase indefinitely. Consequently, the algorithm will converge to the optimal solution through iterations.

After the structure extraction, the snapshot motion-scalar variable estimate $\hat{\mathbf{y}}$ (and the heading vector estimate $\hat{\mathbf{u}}_v = \hat{\mathbf{y}} / |\hat{\mathbf{y}}|$) and angular velocity estimate $\hat{\boldsymbol{\Omega}}$, as well as their covariance matrices, can then be determined using the SVD LS algorithm to solve the linear LS problem (3.4-9) for the given terrain point assignment policy $\{\hat{a}_i\}$. Finally, the snapshot terrain impact time estimates $\{\hat{\tau}_{iz}\}$ and their variances $\{\sigma_i^2\}$can be computed by

$$\hat{\tau}_{iz} = |\mathbf{y}| / \hat{a}_i \text{ and } \sigma_i^2 = |\mathbf{y}|^2\,\mathbf{h}_i^T\mathbb{P}_y\mathbf{h}_i \qquad (3.4\text{-}17)$$

Figure 3.4-3 illustrates the three module architecture of the structure based snapshot estimator: a structure extractor, an egomotion estimator, and an impact time map estimator. The structure extractor takes the image measurements from the image processor, and determines the optimal terrain structure, i.e., which terrain point belongs to which layer. Knowing the terrain structure, the egomotion estimator and impact time map estimator then generate, in parallel, the LS snapshot motion states and impact time map estimation.

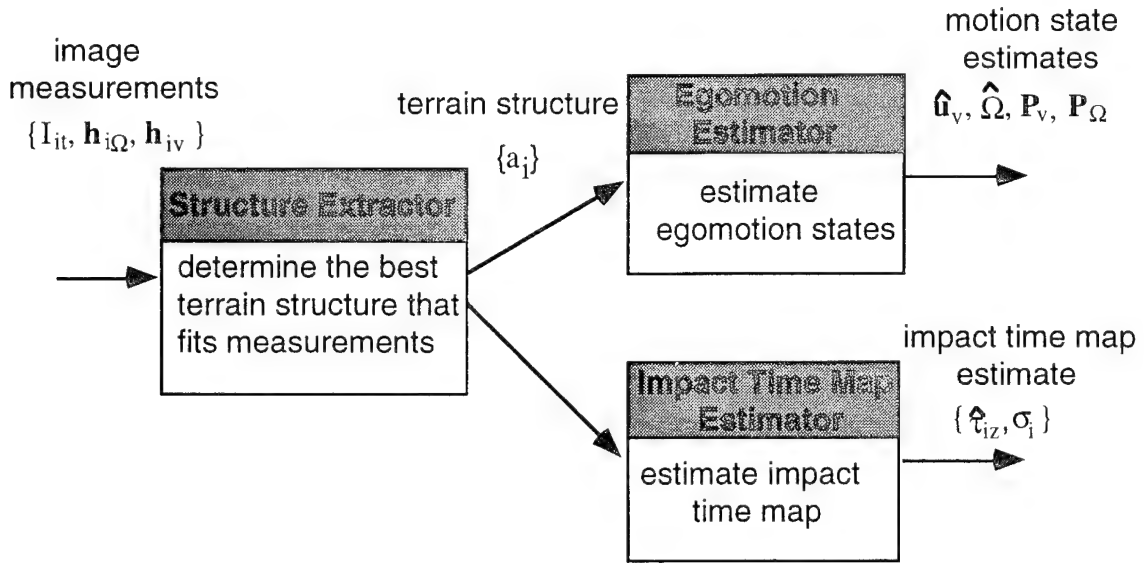


Figure 3.4-3: Structure Extraction Based Snapshot Estimator

The complete structure extraction algorithm is summarized as follows:

Algorithm: Structure Based Snapshot Estimator

1) Initiate
Call SVD algorithm to compute the SVD of $\mathbf{H}_\Omega$ to obtain matrix H and vector $\mathbf{b}_t$
Assign all the terrain points to layer M/2
Compute the matrix $(\mathbf{H}^T\mathbf{H})^{-1}$
2) Make iterative point assignment
Call Point Assignment algorithm *to* determine structure assignments $\{\hat{a}_i\}$
3) Estimate motion state and impact time map
Call SVD LS algorithm to solve (3.4-8) for heading vector $\hat{\mathbf{u}}_v$, angular velocity $\hat{\Omega}$ and their covariances
Generate impact time map $\{\hat{\tau}_i\}$ estimates via (3.4-16)

Algorithm: Point Assignment

For n = 1 to N
Compute coefficients μ_i^n and γ_i^n from (3.4-15a, b)
Determine the optimal assignments between two assignments
Update $[\mathbf{H}^{nT}\mathbf{H}^{nT} + ((\hat{a}_i^{(n+1)})^2 - (\hat{a}_i^n)^2)\mathbf{h}_i\mathbf{h}_i^T]^{-1}$ using Sherman-Morrison Formula given in Appendix B

In actual implementation, we have employed a square-root implementation of the Sherman-Morrison formula to overcome the numerical deficiency of the formula. The structure based snapshot estimator has a computational complexity of the order 9LN flops, where L is the convergence iteration count, N is the number of terrain points.

3.5 Kalman Filter

Figure 3.5-1 shows the architecture of the motion and terrain Kalman filter. Using the snapshot state and terrain impact time estimates as input measurements, we employ a motion state Kalman filter and an impact time map Kalman filter to incrementally improve the motion and terrain shape estimates. Note that the impact time map Kalman filter needs the motion state estimates as its inputs since impact time evolution is specified by the observer motion as explained below in section 3.5.2.

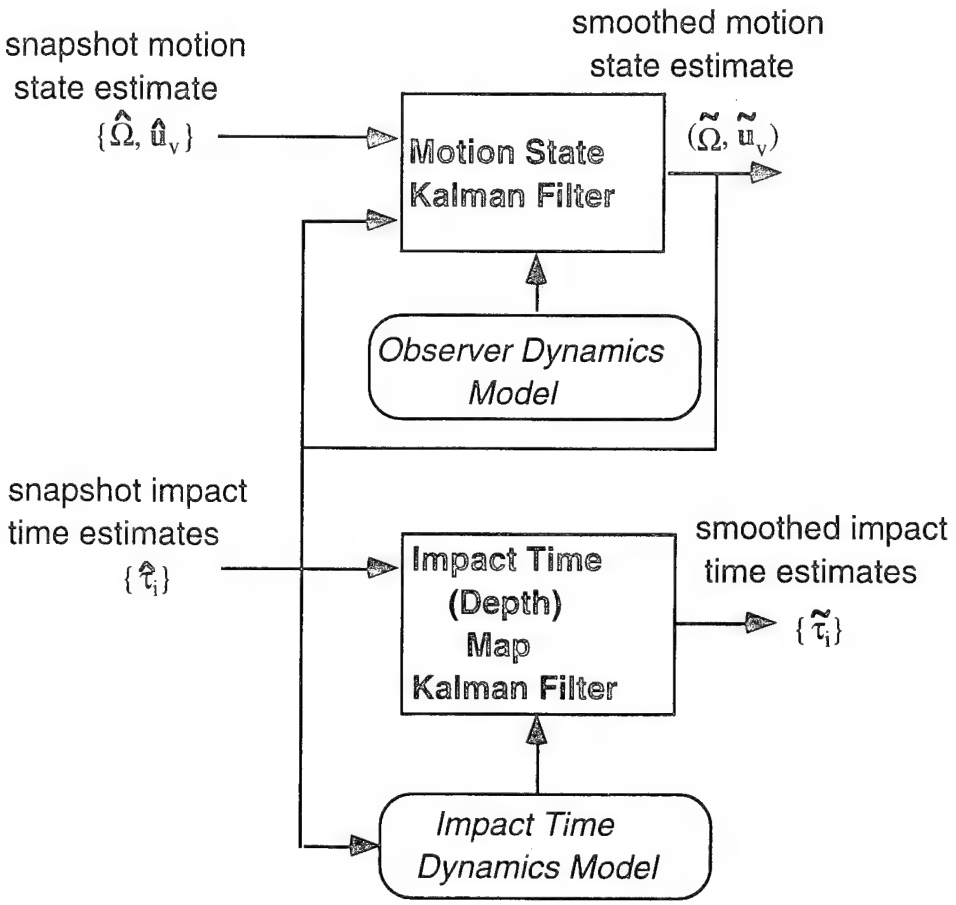


Figure 3.5-1: Kalman Filter

The Kalman filter is a Bayesian estimation technique used to estimate stochastic dynamic system states using noisy observations or measurements. Implementation of a Kalman filter requires specification of three probabilistic models: the system model, the measurement model, and the prior model. The system model describes the evolution of system state over time. The measurement model relates the system state to the observed measurements. The prior model describes the initial knowledge about the system state and its variance before the first measurement is taken. Once the three models are specified, the Kalman filter can then be

implemented using the square root Kalman filter algorithm, presented earlier in section 2.3.

Model complexity is primarily determined by the selection of a coordinate system: the proper system can often transform an apparently intractable filtering problem into one that is readily solvable. For this problem, we choose an observer-fixed coordinate system (i.e., the body coordinate system) since it leads to linear dynamics in both the motion and impact time system models. Once the estimates are determined in the observer-fixed coordinate system, they can then be transformed to another coordinate system via conventional coordinate system transforms. Appendix C gives details of such transforms.

Section 3.5.1 describes the development of the motion state Kalman filter, while section 3.5.2 describes the development of the impact time (depth) map Kalman filter.

3.5.1 Motion State Kalman Filter

We begin by deriving the dynamics equations for the observer's translational and rotational motions. Based on these dynamics equations, we then specify the system and measurement models for the motion state Kalman filter. The specification of the prior models is done implicitly by assuming zero knowledge about state variables before the first measurement.

Let k be the current time (frame) index and $\mathbf{R}_k$ represent the position of the observer at k. For a small frame time Δt, assuming the constant translational velocity between frames we have the translational dynamics equations

$$\mathbf{R}_k = \mathbf{R}_{k-1} + \mathbf{V}_{k-1}\Delta t$$

$$\mathbf{V}_k = \mathbf{V}_{k-1} \qquad (3.5\text{-}1a, b)$$

Remember that we have chosen an observer-fixed coordinate system. In this coordinate system, $\mathbf{R}_k = 0$ since the current position of the observer is always the origin of the coordinate system. Consequently, the selection of the observer-fixed coordinate system reduces the system dynamics equations to (3.5-1b).

The dynamics equation (3.5-1b) can also be written as

$$V_k = V_{k-1}$$

$$\psi_k = \psi_{k-1}$$

$$\gamma_k = \gamma_{k-1} \qquad (3.5\text{-}2a, b, \&c)$$

where $\mathbf{V}_k = V_k(\cos\psi_k \sin\gamma_k, \sin\psi_k \sin\gamma_k, \cos\gamma_k)^T$

The snapshot estimator provides measurements or observations only on the heading vector

$\hat{u}_{v,k}$ which is a unit vector defined by

$$\hat{u}_{v,k} = (\cos \hat{\psi}_k \sin \hat{\gamma}_k, \sin \hat{\psi}_k \sin \hat{\gamma}_k, \cos \hat{\gamma}_k)^T \quad (3.5\text{-}3)$$

where ψ_k is the observer's heading angle and γ_k is the observer's flight path angle γ_k. From the snapshot estimate $\hat{u}_{v,k}$, the heading angle estimate $\hat{\psi}$ and the flight path angle estimate $\hat{\gamma}_k$ can be computed directly via

$$\hat{\psi}_k = \tan^{-1}\left(\frac{\hat{u}_{v,k}(2)}{\hat{u}_{v,k}(1)}\right)$$

$$\hat{\gamma}_k = \cos^{-1}(\hat{u}_{v,k}(3)) \quad (3.5\text{-}4a, b)$$

Consequently, we define a new state variable—heading angle vector $\Phi_k = (\psi_k, \gamma_k)^T$ and specify the translational system and measurement models as

$$\Phi_k = \Phi_{k-1} + \mu_{k-1}$$

$$\hat{\Phi}_k = \Phi_k + \eta_k \quad (3.5\text{-}5a, b)$$

where zero-mean Gaussian random variables μ_k and η_k account for the unmodeled dynamics and observation errors, respectively.

We now derive the dynamics equations for the observer's rotational motion. There are many different ways to describe an observer's rotational motion (e.g. the nine-parameter expression, the quaternion expression, the Rodrigures expression, and the conventional Euler angle roll-pitch-yaw expression). Each expression leads to a different system model. We select Eulerangle roll-pitch-yaw expression that uses three consecutive rotations around three principle axes to characterize an arbitrary rotation. Let $\Theta = (\theta_x, \theta_y, \theta_z)^T$ be the three Euler angles—yaw, pitch, and roll angles defining the rotations around the three axes (X, Y, Z) of a given coordinate system. The corresponding rotation matrix $\mathbb{T}$ is given as follows:

$$\mathbb{T} = \mathrm{RPY}(\Theta) \equiv \mathrm{rot}(\theta_z)\, \mathrm{rot}(\theta_y)\, \mathrm{rot}(\theta_x) \quad (3.5\text{-}6a)$$

where

$$\mathrm{rot}(\theta_x) = \begin{bmatrix} \cos\theta_x & \sin\theta_x & 0 \\ -\sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{rot}(\theta_y) = \begin{bmatrix} \cos\theta_y & 0 & -\sin\theta_y \\ 0 & 1 & 0 \\ \sin\theta_y & 0 & \cos\theta_y \end{bmatrix}$$

$$\text{rot}(\theta_z) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_z & \sin\theta_z \\ 0 & -\sin\theta_z & \cos\theta_z \end{bmatrix} \qquad \text{(3.5-6b, c, \&d)}$$

Let the matrix $\mathbf{T}_k$ represent the observer's current orientation specified using the roll-pitch-yaw expression. For a small frame time Δt, assuming the constant rotation angles $\boldsymbol{\theta}_k$ between frames, we have the following rotation dynamics equations:

$$\mathbf{T}_k = \mathbf{T}_{k-1}\,\text{RPY}(\boldsymbol{\theta}_{k-1})$$

$$\text{RPY}(\boldsymbol{\theta}_k) = \text{RPY}(\boldsymbol{\theta}_{k-1}) \qquad \text{(3.5-7a, b)}$$

Remember again that we have chosen an observer-fixed coordinate system. In this coordinate system, $\mathbf{T}_k = \mathbf{I}$ since the current orientation of the observer defines the observer's body coordination system. Consequently, the selection of the observer-fixed coordinate system reduces the dynamics equations to (3.5-7b).

Although the dynamics equation (3.5-7b) completely define the rotational motion, it is not a system model since the nine-element rotation matrix $\text{RPY}(\boldsymbol{\theta}_k)$ is uniquely specified by the three-variable rotation angles

$$\boldsymbol{\theta}_k = \boldsymbol{\Omega}_k \Delta t \qquad \text{(3.5-8)}$$

Let RPY^{-1} be the inverse operator of RPY. That is, RPY^{-1} takes a rotation matrix as the operand and returns its Euler angles. Applying RPY^{-1} on both sides of (3.5-7b) and canceling the constant Δt, we have the rotation system model, given by

$$\boldsymbol{\Omega}_k = \boldsymbol{\Omega}_{k-1} + \boldsymbol{\varepsilon}_{k-1} \qquad \text{(3.5-9a)}$$

where zero-mean Gaussian random variables $\boldsymbol{\varepsilon}_k$ accounts for the unmodeled dynamics. We specify the rotation measurement model as

$$\hat{\boldsymbol{\Omega}}_k = \boldsymbol{\Omega}_k + \boldsymbol{\zeta}_k \qquad \text{(3.5-9b)}$$

where zero-mean Gaussian random variables $\boldsymbol{\zeta}_k$ account for the observation errors.

Using the snapshot angular velocity and heading angle estimates as the measurements, the translational and rotational system models of (3.5-5a) and (3.5-9a), and measurement models o of (3.5-5b) and (3.5-5c), we can then directly use the square root Kalman filter algorithm presented earlier in section 2.2.3 to generate the smoothed motion state estimates.

3.5.2 Terrain Impact Time Map Kalman Filter

The impact time map Kalman filter generates the smoothed impact time estimates using the

snapshot estimates of the impact time as measurements. We adopt a pixel-based (iconic) approach to the Kalman filter implementation, where the impact time at a pixel is defined as the state variable.

Remember that we have chosen the body coordinate system for the system dynamics description. In this coordinate system, assuming a rigid terrain surface, the system model for the ith pixel impact time (state variable) can be expressed as

$$\tau_{i,k} = \tau_{i,k-1} + \varepsilon_{k-1} \tag{3.5-10}$$

where both $\tau_{i,k}$ and $\tau_{i,k-1}$ are represented in the observer-fixed coordinate system at time (frame) k, and ε_{k-1} is a zero-mean Gaussian noise accounting for the model error. The measurement model is given by

$$\hat{\tau}_{i,k} = \tau_{i,k} + \xi_k \tag{3.5-11}$$

where ξ_k represents the measurement error.

The snapshot estimator provides $\hat{\tau}_{i,k}$ and its variance $\sigma_{i,k}$. Given the system and measurement models, the implementation of the Kalman filter for impact time smoothing is straight forward except the computation of estimate $\hat{\tau}_{i,k}^-$ in the prediction phase according to (2.2-17). The available estimate $\hat{\tau}_{i,k-1}$ is made in the body coordinate system at time k-1 while the computation of $\hat{\tau}_{i,k}^-$ needs the estimate made in the body coordinate system at time k.

We develop a numerical extrapolation approach to generating the needed estimate of $\hat{\tau}_{i,k}^-$ at time k, using the available impact time map estimates $\{\hat{\tau}_{i,k-1}\}$ at k-1 and the known observer motion states $\hat{\Omega}_{k-1}$ and $\hat{u}_{v,k-1}$. Figure 3.5-2 illustrates the impact time evolution between two consecutive frames. The evolution is determined by two key factors: relative motion of the observer with respect to the terrain and a finite field-of-view (FOV) limiting the observer's instantaneous view of the terrain. Specifically, the relative motion implies that $\hat{\tau}_{i,k}$ will change from one image frame to another since it is measured relative to the observer. The limited FOV implies that some new imagery (thus its impact times) will enter the impact time map and some old images will move out of the map.

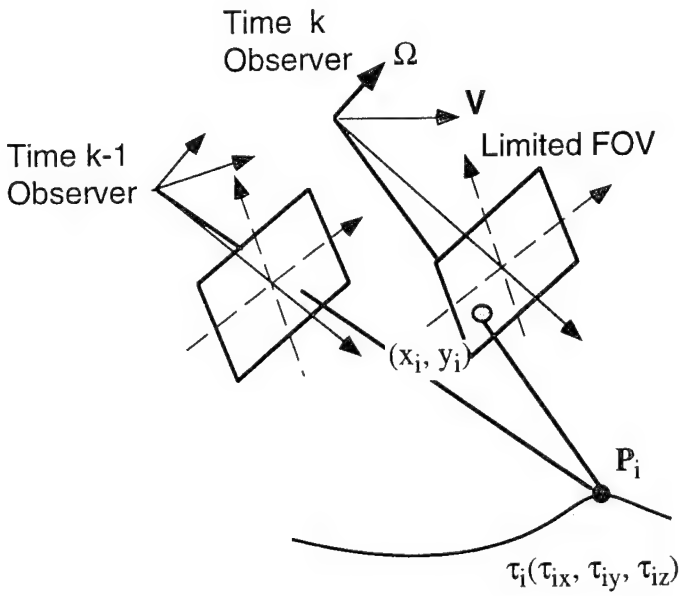


Figure 3.5-2: Observer Motion Changes

At time k-1, the estimated terrain point i's impact time vector is given by

$$\hat{\boldsymbol{\tau}}_{i,k-1} = (\hat{\tau}_{iz,k-1}x_i, \hat{\tau}_{iz,k-1}y_i, \hat{\tau}_{iz,k-1})^T \tag{3.5-12}$$

from the perspective projection. At time k, the observer and his associated coordinate system move to a new location. In the observer-fixed coordinate system, the orientation change due to rotations can be determined from equation (3.5-7a) and is given by

$$\mathbf{T}_{k-1} = [\mathrm{RPY}(\hat{\boldsymbol{\Omega}}_{k-1}\Delta t)]^T \tag{3.5-13}$$

The impact time change due to the translational motion can be determined from equation (3.5-1a) by normalizing the equation using the observer speed V_{k-1}

$$\boldsymbol{\tau}_{k-1} = -\hat{\mathbf{u}}_{v,k-1}\Delta t \tag{3.5-14}$$

according to equation (3.5-2b).

Given the estimated impact time vector $\hat{\boldsymbol{\tau}}_{i,k-1}$, the angular velocity estimate $\hat{\boldsymbol{\Omega}}_{k-1}$, and the heading vector estimate $\hat{\mathbf{u}}_{v,k-1}$, the impact time vector at time k can be estimated as

$$\hat{\boldsymbol{\tau}}_k^i = [\mathrm{RPY}(\hat{\boldsymbol{\Omega}}_{k-1}\Delta t)]^T[\hat{\boldsymbol{\tau}}_{i,k-1} - \hat{\mathbf{u}}_{v,k-1}\Delta t] \tag{3.5-15}$$

where the superscript i denotes that $\hat{\boldsymbol{\tau}}_k^i$ is computed using the impact time vector $\hat{\boldsymbol{\tau}}_{i,k-1}$ of pixel i at time k-1. Note that because of the observer motion and the limited horizon, the estimated impact time vector $\hat{\boldsymbol{\tau}}_k^i$ would not project to pixel *i* at *k*, but to new image coordinates $(\hat{x}_k^i, \hat{y}_k^i)^T$, determined by

$$(\hat{x}_k^i, \hat{y}_k^i)^T = (\frac{\hat{\tau}_{x,k}^i}{\hat{\tau}_{z,k}^i}, \frac{\hat{\tau}_{y,k}^i}{\hat{\tau}_{z,k}^i})^T \text{ for } i = 1, 2, ..., N \quad (3.5\text{-}16)$$

As shown in figure 3.5-3, the closest pixel row index $\hat{m}$ and column index $\hat{n}$ for the image coordinates $(\hat{x}_k^i, \hat{y}_k^i)^T$ can be computed, respectively, by

$$\hat{m} = int[(\frac{\hat{x}_k^i N_\Psi}{\tan(\Psi / 2)} + N_\Psi + 1) / 2] \quad (3.5\text{-}17)$$

$$\hat{n} = int[(\frac{\hat{y}_k^i N_\Theta}{\tan(\Theta / 2)} + N_\Theta + 1) / 2] \quad (3.5\text{-}18)$$

where int(d) is an integer operator which returns an integer that is closest to the value d, Θ is the vertical FOV angle, N_Θ is the number of the vertical pixels, Ψ is the horizontal FOV angle, and N_Ψ is the number of horizontal pixels.

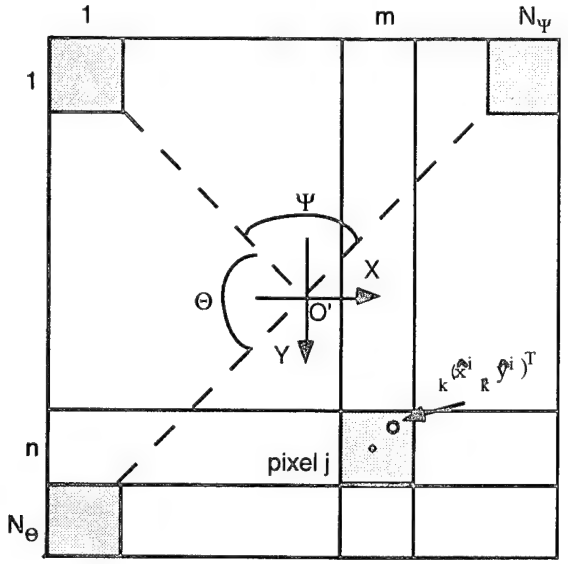


Figure 3.5-3: Image Coordinates

As shown in figure 3.5-3, an image point $(\hat{x}_k^i, \hat{y}_k^i)^T$ either stays within the image FOV at k if $1 \le \hat{m} \le N_\Psi$ and $1 \le \hat{n} \le N_\Theta$, or moves out if one of the inequality constraints does not hold. In the first case, we consider pixel $j = (\hat{n} - 1)N_\Psi + \hat{m}$ at k has an impact time $\hat{\tau}_k^i$ and a variance $P_{j,k}^-$, computed in the prediction phase of the Kalman filter by

$$\hat{\tau}_{j,k}^- = \hat{\tau}_k^i \quad (3.5\text{-}19a)$$

$$P_{j,k}^- = P_{i,k} / \alpha \quad (3.5\text{-}19b)$$

where $\alpha \le 1$ is a discount factor accounting for the uncertainty in impact time estimation via the numerical extrapolation.

4. SYSTEM DEMONSTRATION AND EVALUATION

This chapter demonstrates and evaluates overall operation of the ATAS. We show that the ATAS system can accurately generate navigation critical observer motion state and terrain shape information, such as observer heading, observer angular velocity, and the observer impact time (depth) map across all pixels. Section 4.1 describes the image generation and defines the performance evaluation criterion. Section 4.2 presents evaluation results using flat terrain CGI, section 4.3 presents the evaluation results using non-flat terrain, and section 4.4 presents the evaluation results using flight-recorded and real-time frame-grabbed terrain imagery. Section 4.5 concludes the chapter with a summary of the evaluation results.

4.1. Image Generation and Performance Criterion

Figure 4.1-1 illustrates the overall architecture of the image generator, which converts computer generated imagery (CGI) or real imagery generated by a (digital or video) camera into a sequence of digitized (gray level) image frames. As shown in figure, images, when generated in analog format either via a computer or a video camera, must first pass through a frame grabber to be grabbed into digitized image frames.

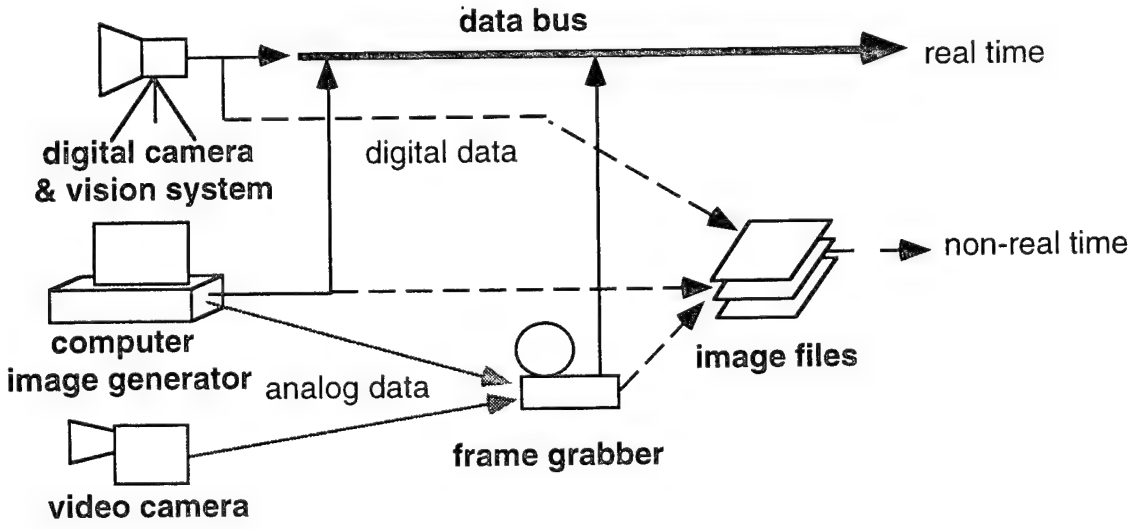


Figure 4.1-1: Image Generator

The image generator has two operating modes: real-time and non-real time, as shown in figure 4.1-1 via the solid and dashed lines, respectively. In the real-time mode, the image generator sends the digitized images directly to the estimator at a fixed frame rate for real-time estimation. Consequently, in the real-time mode, the estimator must be able to generate each estimate before the next frame arrives. In the non-real time mode, the image generator stores the

digitized images as image files and the estimator then reads the digitized image frames from image files, frame by frame. The estimator can thus take time to finish an estimate before it reads the next image frame from the image files.

When real imagery is generated using a digital camera or when CGI is generated in digital format in real-time, real-time operation of ATAS then depends only on the estimator's host processor being fast enough to finish each estimate before the next image arrives. When real or CGI imagery is generated in analog format, whether the system can operate in real or non-real time mode then also depends on whether or not the frame grabber is capable of real-time image grabbing.

The image generator provides both the CGI and real imagery for on-line motion and terrain shape estimation. For CGI generation, we developed and implemented a flat-terrain image generator that produces CGI frames simulating perspective images viewed by an observer (camera) flying over a flat terrain decorated by arbitrary sinusoidal patterns. We also developed and implemented a rolling-terrain image generator that produces video (analog) imagery simulating images viewed by an observer flying over a sinusoidal rolling terrain decorated by sinusoidal patterns. In addition, from various sources we obtained other complex CGI terrain images used widely for computer vision algorithm testing. For real image generation, we obtained video imagery using a video camera fixed on a helicopter flying over real terrain. To convert the analog video imagery into the digitized image frames, we implemented a frame grabber using hardware and software frame grabbing tools hosted on a Macintosh Quadra 840 AV computer.

Section 4.1.1 briefly describes the CGI generation process, section 4.1.2 the real image generation process, and section 4.1.3 the image grabbing process. Section 4.1.4 describes the performance evaluation criterion.

4.1.1 CGI Generation

Figure 4.1-2 is an overview block diagram of the **flat terrain image generator**, which can either produce image data files to support the non-real time evaluation of the estimator, or generate digitized image frames at a fixed frame rate to support real-time evaluation, if the image generator and estimator's host processor(s) has sufficient processing power.

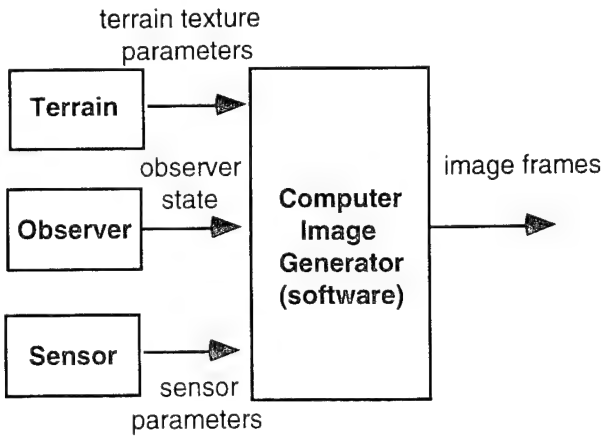


Figure 4.1-2: Flat Terrain Image Generator

As shown in the figure, three modules drive the flat-terrain computer image generation: Terrain, Observer, and Sensor. The terrain module defines the texture pattern on the ground plane, to be imaged by the observer (sensor). It supports fairly complex pattern generation, in which pattern intensity is a two-dimensional sum-of-sinusoids, of variable spatial frequency, intensity amplitude, and spatial phase shift. This capability supports the generation of texture patterns which span the range of appearance from perfectly regular to totally random.

The Observer module simulates the kinematics involved in the observer's flying over the ground terrain. The module generates the terrain-relative observer states of position ($\mathbb{R}$), velocity ($\mathbb{V}$), attitude ($\mathbb{A}$), and angular velocity (Ω), and updates them every simulation time step.

The Sensor module simulates the basic geometric attributes of the imaging observer's sensor geometry configuration. It specifies the sensor boresight in the vehicle body-axis coordinates ($\mathbf{u}_S^B$), as well as the size of the FOV (Ψ_{Fov}, Θ_{Fov}) and the lateral and vertical pixel count (N_Ψ, N_Θ). This module supports arbitrary boresighting and orientation of the sensor FOV, for general configurations.

The terrain texture parameters, observer states, and sensor parameters are then combined by a computer image generator module: a computer algorithm that generates a sequence of image frames in true perspective, as seen by the observer in flight over simulated terrain. Each image frame is defined by a set of N intensity scalars $\{I_i\}$, one for each of the N sensor pixels. One image frame is produced every simulation time tick. The computation algorithm of the flat terrain image generation is implemented in the C programming language.

Figures 4.1-3 to 4.1-5 illustrate a group of image frames generated by the flat terrain image generator. Figure 4.1-3a to 4.1-3d show images with a single sinusoidal of 500 ft wavelength

decoration in both x and y directions, a field-of-view (FOV) of 32° x 32°, and an observer flying straight-and-level at an altitude of 200 ft and a speed of 300 ft/second with different viewing depression angles. The images at the left are generated at a high resolution of 128 x 128 pixels per image frame, and the images at the right are generated at a low resolution of 32 x 32 pixels per image frame.

As shown in figure 4.1-3a, at zero boresight depression angle and with a resolution of 128 x 128 pixels, the generated image approximates the true unpixellated terrain image viewed through the sensor FOV. The upper half is sky, and has a neutral grey value. The lower half shows the terrain pattern, which naturally recedes to the horizon in the 2D sinusoidal pattern used to "decorate" the terrain. With a low resolution of 32 x 32 pixels, the effects of pixellation on the image are clearly evident, as shown in figure 2.2b.

As shown in figure 4.1-3c, at a larger boresight depression angle of 45°, the horizon is no longer visible. A "flatter" 2D sinusoidal pattern appears in the image. Similar pixellation effects are also clearly evident.

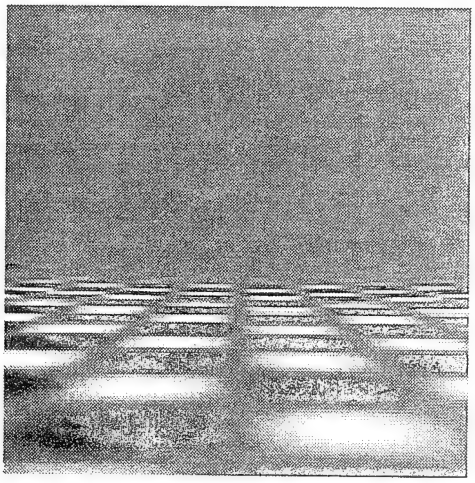


Figure 4.1-3a: High Resolution Image at Zero Depression Angle

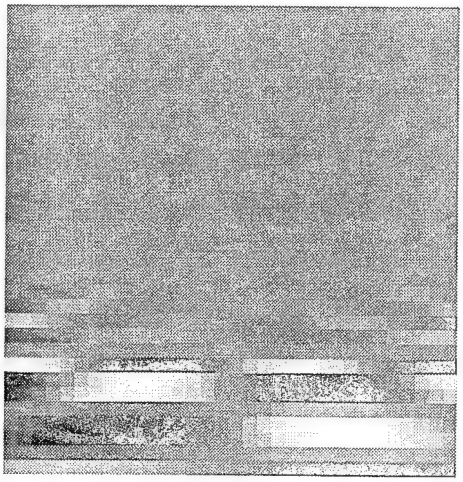


Figure 4.1-3b: Low Resolution Image at Zero Depression Angle

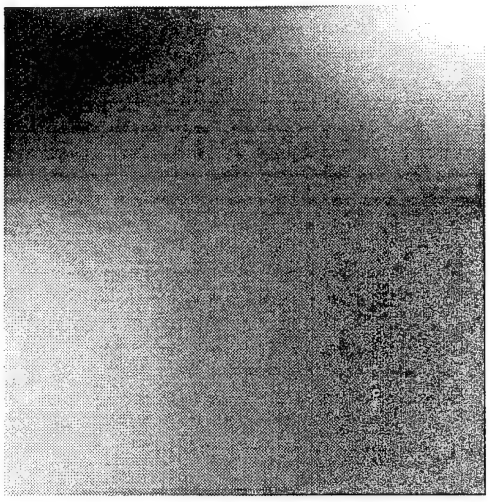


Figure 4.1-3c: High Resolution Image at 30° Depression Angle

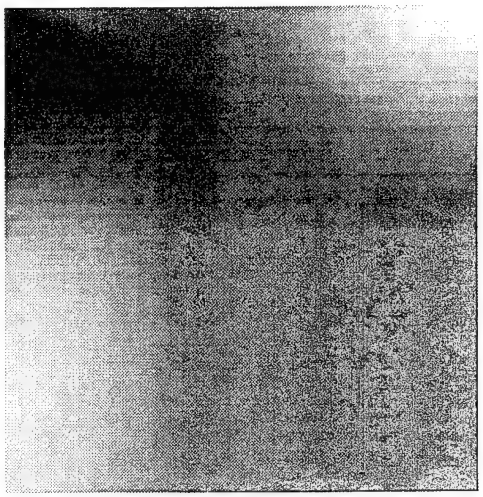


Figure 4.1-3d: Low Resolution Image at 30° Depression Angle

Figures 4.1-4a and 4.1-4b show 45° depression angle images at an FOV of 64° x 64°, at high and low resolution, respectively. Comparing with images 4.1-3c and 4.1-3d, we see the effects of FOV on viewing more decorated 2D sinusoidal patterns.

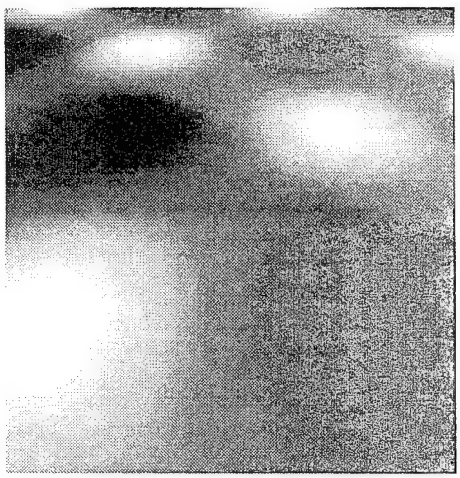


Figure 4.1-4a: High Resolution Image at 40° Depression Angle & 65° FOV

Figure 4.1-4b: Low Resolution Image at 30° Depression Angle & 65° FOV

Figures 4.1-5a and 4.1-5b show 45° depression angle images, at a shorter wavelength of 50 feet, at high and low resolutions, respectively. Comparing with images 4.1-3c and 4.1-3d, we see clearly the effects of wavelength on the decorated 2D sinusoidal patterns of the viewed image.

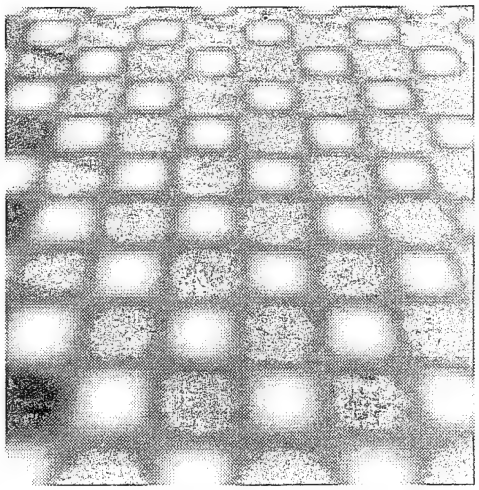


Figure 4.1-5a: High Resolution Image at 45° Depression Angle & 50 ft Wavelength

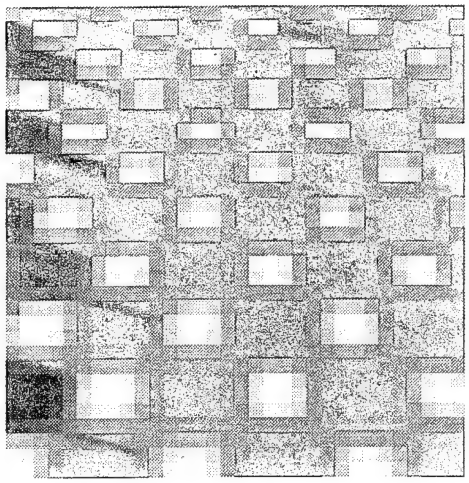


Figure 4.1-5b: Low Resolution Image at 45° Depression Angle & 50 ft Wavelength

Figure 4.1-6 illustrates an overview block diagram of the **rolling terrain image generator** developed on a Silicon Graphics (SG) computer (SG-Indigo R4000 Elan). Similar to the flat terrain image generator, the rolling terrain image generator employs a terrain module to define the terrain texture pattern, an observer module to simulate observer kinematics involved in the observer's flying over the ground terrain, and a sensor module to specify the sensor geometry configuration. The terrain module also defines rolling terrain topography approximated using sinusoidal functions. The rolling terrain image generator does not directly produce digital image frames, but instead uses built-in graphics engine (implemented in hardware) to display the perspective view of textured rolling terrain, that is subsequently recorded on videotape.

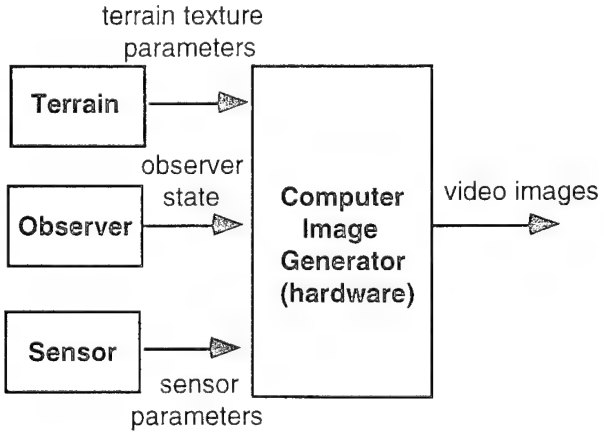


Figure 4.1-6: Rolling Terrain Image Generator

Figure 4.1-7 shows a frame of a rolling terrain image sequence that was first generated by the

rolling terrain image generator and then grabbed into a single image frame by the frame grabber.

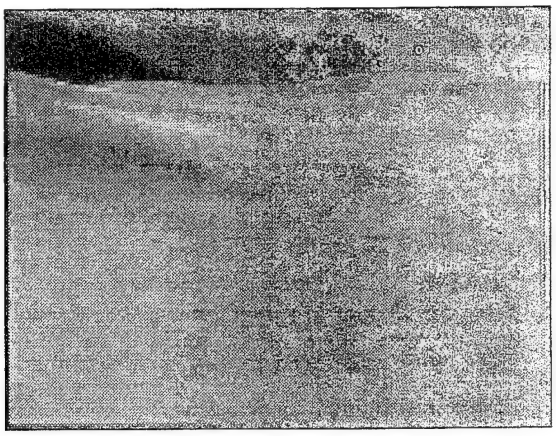


Figure 4.1-7: Rolling Terrain Images

Besides this type of simple structure (flat and rolling) terrain imagery, we obtained frames of the famous Yosemite image sequences that contain very complicated terrain structures for system evaluation (Barron et al., 1994). Figure 4.1-8 shows one frame of the Yosemite sequence simulating an observer (camera) with a 40° FOV flying through Yosemite valley. The heading vector for the image sequences is $(0, 0.17, 0.98)^T$, the rotation axis is $(0.14, 0.98, 0.17)^T$, and the rotation rate is 0.095°/frame* . The frame size of the images are 316 by 252.

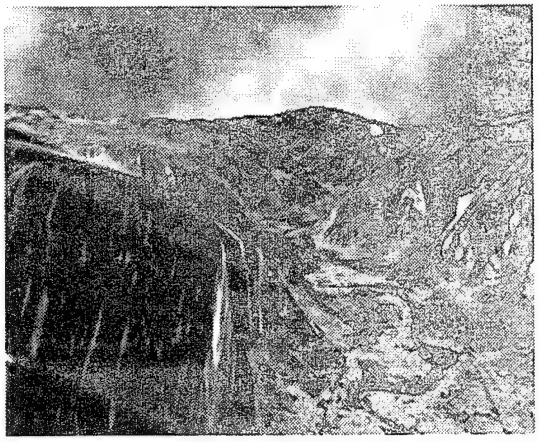


Figure 4.1-8: An Image Frame from the Yosemite Sequence

4.1.2 Real Imagery Generation

We obtained from the NASA Ames Research Center flight-recorded imagery taken by a

---

* where x is in the horizontal direction, y is in the vertical direction, and z is into the image plane.

camera fixed on a helicopter flying over an airfield. The helicopter flies straight and level at a speed around 32 feet/sec and an altitude 16 feet above the ground. The camera boresight angle is -8° , and the lateral and vertical FOV are 42° and 44°, respectively. The image sequences have 40 image frames, each having a size of 512 x 477 pixels. The imagery is kept in two formats: image frame data files and video tapes. Figure 4.1-9 shows a frame from the sequence.

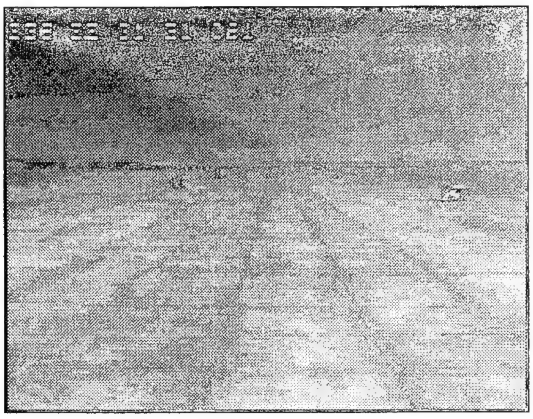


Figure 4.1-9: An Image Frame from the NASA Image Sequences

4.1.3 Frame Grabber

Figure 4.1-10 illustrates a real-time implementation of the image generator. Without a real-time digital camera system, our effort concentrated on real-time frame grabbing of video images. As shown in the figure, an image video tape was first generated either using a camcorder to take pictures in flight or using the rolling terrain image generator described in section 3.1. The tape could then be played back to generate the video images in the ground-based ATAS environment. Finally, a frame grabber was implemented for real-time frame grabbing.

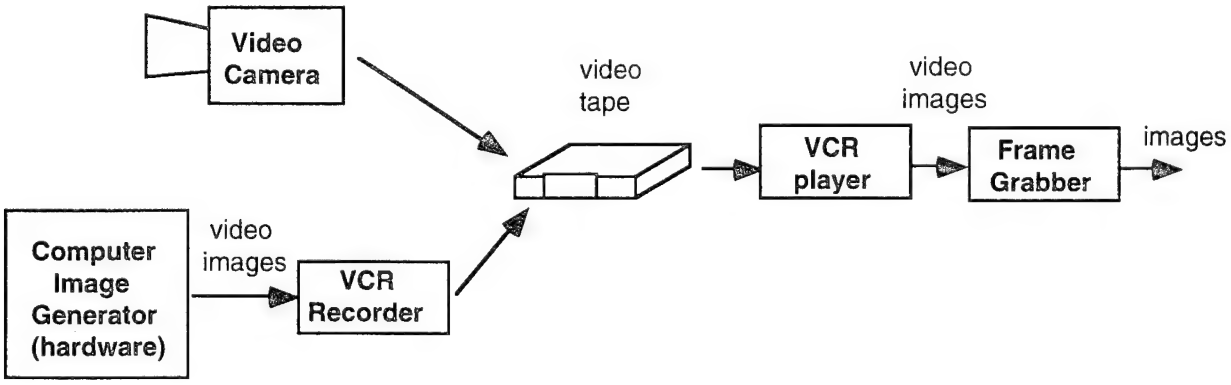


Figure 4.1-10: Real-Time Image Generator Implementation

We implemented the frame grabber on a Macintosh Quadra 840 AV. We chose the Macintosh Quadra 840 AV computer because it has a fast 40 MHz microprocessor, 38 MB

RAM, built-in digital audio and video (DAV) hardware, and an image compression board— the SpigotPower AV that supports real-time (30 Hz) frame grabbing. Using the frame grabbing software tool Video Fusion 1.5.1, the AV computer accepts image signals from a video cassette recorder (VCR), digitizes them, and then outputs the digitized image frames at a user-specified sampling rate. For non-real time operation, we first captured the grabbed image frames in computer RAM and then stored them into image files. For real-time operation, the grabbed image frame is sent directly to the estimator running on the same or another computer.

Figures 4.1-11a shows a frame of the grabbed image. Compared with the ideally generated image frame shown in 4.1-11b, deterioration of image quality during the image grabbing process is evident.

Figure 4.1-11a: Grabbed Image

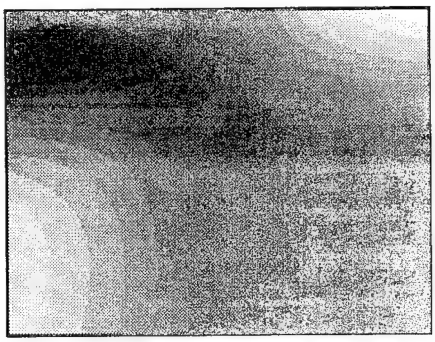


Figure 4.1-11b: Ideally Generated Image

The major problem with real-time frame grabbing, however, is not that the images cannot be grabbed at high quality, but that the frame grabber does not support flicker-free frame grabbing. Two problems persist. The first is that the time (rate) needed for grabbing an image depends on the contents of the image. Consequently, each image is grabbed at a different rate since all images are different. In addition, the frame grabber does not provide information on the individual frame grabbing rate. Without a constant grabbing rate or knowledge of the individual grabbing rate, it is impossible for the egomotion and terrain shape estimator to properly integrate estimates from multiple images in the Kalman filtering process.

The second problem, the more severe one, is that the frame grabber has an unwanted adaptation function that is built into its frame grabbing process. If the frame grabber does not complete a single frame grab within the user specified frame time, it either automatically drops the incoming image frame, outputting the result from the previous frame as the current one, or grabs a partial image from the previous frame, continuing from where it stops on the current one. This adaptation function adds considerable noise to the grabbed image sequence.

It should be note, however, that neither problem occured with the frame grabbed image

sequence obtained by NASA under real-time recording conditions. Thus, for ATAS testing, the images would be considered as eithe generated in real-time (via direct sensor feed to the estimator) or as recorded and subsequently frame-grabbed in real-time.

4.1.4 Performance Criterion

To evaluate system performance for both the computer generated and real images, several performance indices are defined. These include:

- *Aimpoint Error:* This is obtained from the angular difference between true heading $\mathbf{u}_v$ and heading $\hat{\mathbf{u}}_v$ estimated by ATAS. The aimpoint error reflects total aim estimation error in both vertical and lateral directions, and is computed by

$$\varepsilon_{aim} = \cos^{-1}(\mathbf{u}_v \circ \hat{\mathbf{u}}_v) \tag{4.1-1}$$

- *Angular Rate Error*: This is obtained from the magnitude of the difference between actual angular velocity Ω and angular velocity $\hat{\Omega}$ estimated by the system. The angular rate error reflects total body-rate error across all three body axes, and is computed by

$$\varepsilon_{arg} = \left|\Omega - \hat{\Omega}\right| \tag{4.1-2}$$

- *Impact Time Error:* This is obtained by first determining the difference between the true pixel-dependent terrain-surface impact time τ_i with the corresponding estimate $\hat{\tau}_i$ generated by the ATAS system, at each visible ground pixel, and then averaging the results. Note that when the impact time is computed in the navigation system, or in the body coordinate system of an observer undergoing straight and level flight, the average impact time equals the observer's altitude normalized by his speed. Assuming that the system has access to an accurate estimate of the observer speed v, this error index reflects the precision of the system in estimating the effective altitude (range) of the observer above the terrain, in time units.

- *Across-Frame Root-Mean Square (RMS) Error:* This is obtained by first computing the error mean μ and stand deviation s across multiple frames. The RMS error is then computed via

$$\text{RMS} = \sqrt{\mu^2 + s^2} \tag{4.1-3}$$

4.2 Evaluation Using Flat Terrain CGI

Two kinds of evaluations using the flat terrain CGI were performed to examine the performance of ATAS: the evaluation of system estimation accuracy and the evaluation of the system for modeling the human flow-based egomotion process. These evaluations are presented

in sections 4.2.1 and 4.2.2, respectively.

4.2.1 Estimation Accuracy with Flat Terrain CGI

A sequence of images was computer generated using the flat-terrain image generator illustrated in figure 3.2-1. The simulated vehicle flies straight-and-level with a speed of 300 ft/sec at an altitude of 200 feet above the terrain, with zero body attitude rates. The vehicle sensor was boresighted 15° below the horizon. The sensor field-of-view (FOV) was set to 32° laterally by 32° vertically, and the pixel count was 32 laterally by 32 vertically to yield a nominal 1° by 1° pixel. The simulated terrain is flat and featureless and is textured via a spatial 2D sinusoidal pattern with a wavelength of 500 feet in both vertical and lateral directions.

Figure 4.2-1 shows one image frame generated under the above conditions. With a resolution of 128 x 128 pixels, the figure approximates the true unpixellated terrain image seen through the sensor field-of-view. The upper 10% of the figure is sky, and has a neutral gray value. The lower portion below the horizon shows the terrain pattern, which naturally recedes to the horizon. Note the regular 2D sinusoidal pattern used to "decorate" the surface. Figure 4.2-2 illustrates the image actually *imaged* by the sensor at the 32 x 32 pixel resolution. The effects of pixellation are clearly evident.

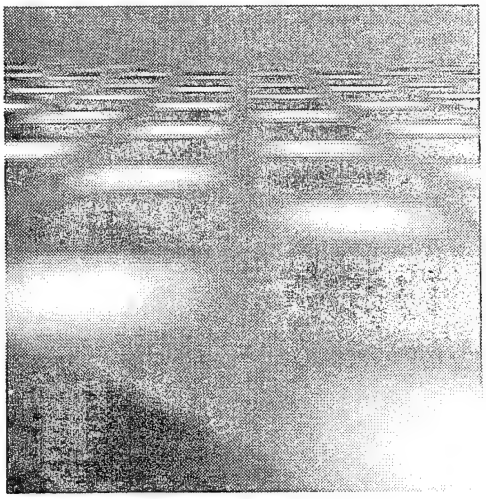


Figure 4.2-1: High Resolution Image

Figure 4.2-2: Low Resolution Image Used for Estimation

In the simulation, a sequence of consecutive images like these are generated, and then processed, frame by frame, by ATAS to obtain the state and terrain shape estimates. For the simulations described in this section, the interframe sample time is set at 0.02 sec (a frame rate of 50 Hz), so that an imaged terrain point in the center of the sensor focal plane will move at about

one pixel/frame, at the given nominal altitude, speed, and boresight angle.

Using the same image sequences, we first evaluated the geometry based egomotion and terrain shape estimator (section 4.2.1.1), and then the structure based estimator (section 4.2.1.2).

4.2.1.1 Geometry Based Estimator

Figures 4.2-3 to 4.2-6 illustrate the motion state and terrain parameter estimation time histories for ATAS using the geometry based estimator. The system used a flat terrain model, and assumed an initial +5° error in both the flight path and the heading angles to reflect the initial uncertainty on the observer motion state. The model standard deviations for heading and path angles were set at 0.01°, the model variances for angular velocity vector were set at 0.01 °/sec^2 for each individual vector element. The discount factor for terrain impact time (range) map evolution was set at 0.9.

In the figures, the solid line represents the snapshot estimates, the thick solid line represents the Kalman filter smoothed estimates, and the dashed line represents the true observer state and terrain shape parameters.

Figure 4.2-3 shows a time history for the estimates of the observer impact time (altitude/speed), which is the cross-pixel average of the impact time map in the body coordinate system. As shown, both the snapshot and filtered impact times oscillate around the actual impact time. Note that a larger variation occurs in the snapshot estimate, and the effects of the Kalman filter in smoothing the oscillation of the snapshot estimate and improving estimation accuracy are clearly demonstrated. Oscillation in the estimate is due to the periodic nature of the terrain sinusoidal pattern being viewed.

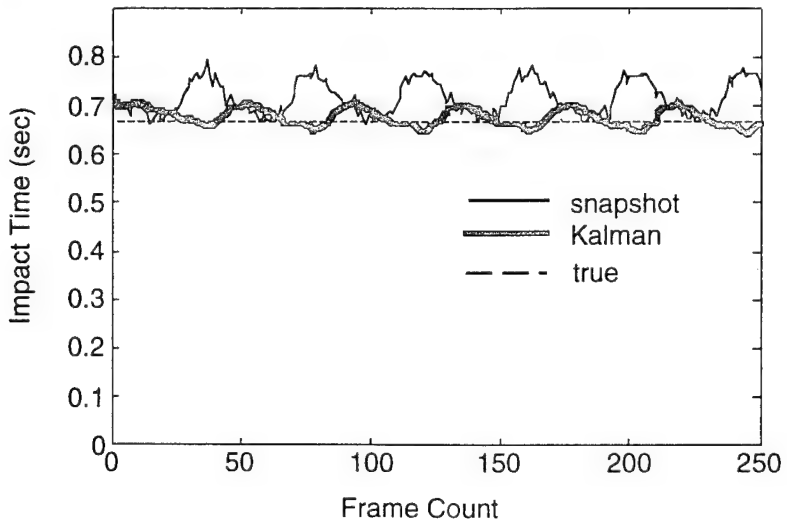


Figure 4.2-3: Impact Time Estimation History

Figures 4.2-4 and 4.2-5 illustrate time histories of the estimated heading and flight path angles, respectively. As illustrated, the snapshot heading angle estimate oscillates within a 0.5° range around the true heading vector, while the snapshot flight path angle estimate oscillates within a 1.5° range with about a 1.5° bias. The filtered heading angle converges to the true heading angle, while the filtered flight path angle converges with about a 1.5° bias.

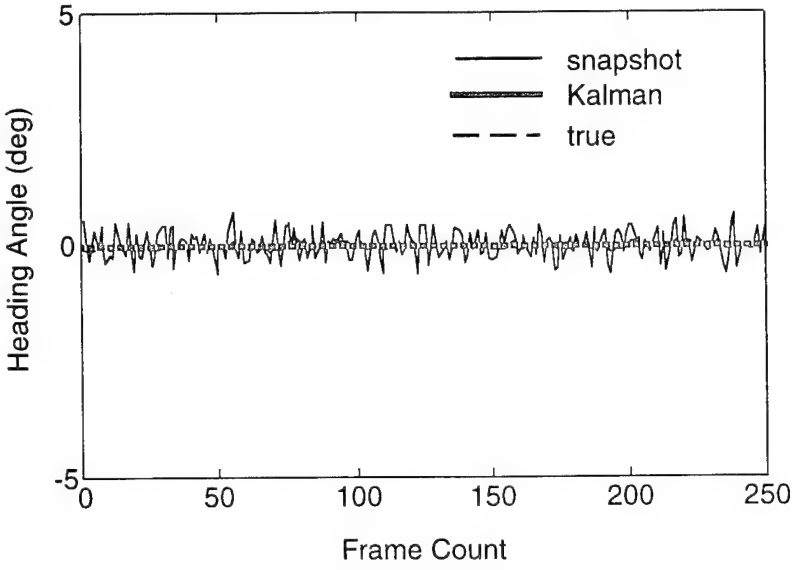


Figure 4.2-4: Heading Angle Estimation History

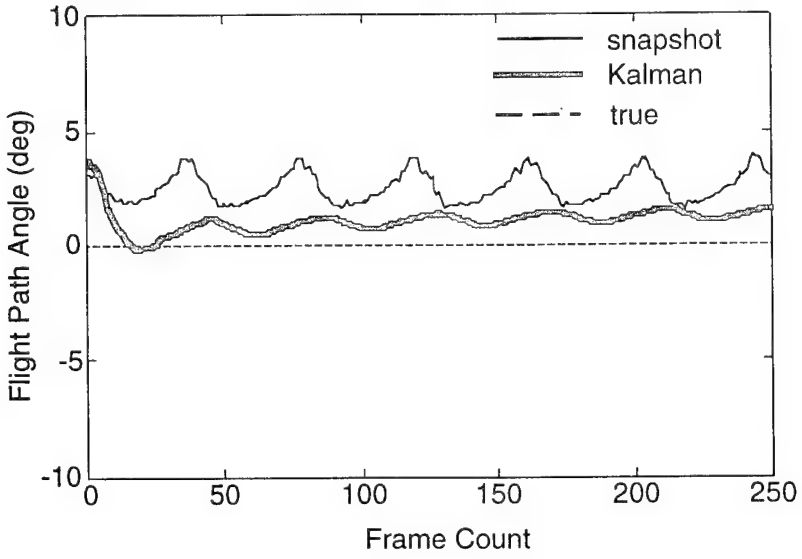


Figure 4.2-5: Flight Path Angle Estimation History

Figure 4.2-6 shows a time history of the estimated pitch (y-axis) angular velocity. As shown, both the snapshot and filtered estimates oscillate with a bias of 2.5°. Again, the effects of Kalman

filtering in smoothing the oscillation of the snapshot estimate and improving estimation accuracy are clearly demonstrated. The x and z-axis angular velocity estimates are not shown, since they estimated the true anagular rates (which are zero) with errors of less than 0.01°.

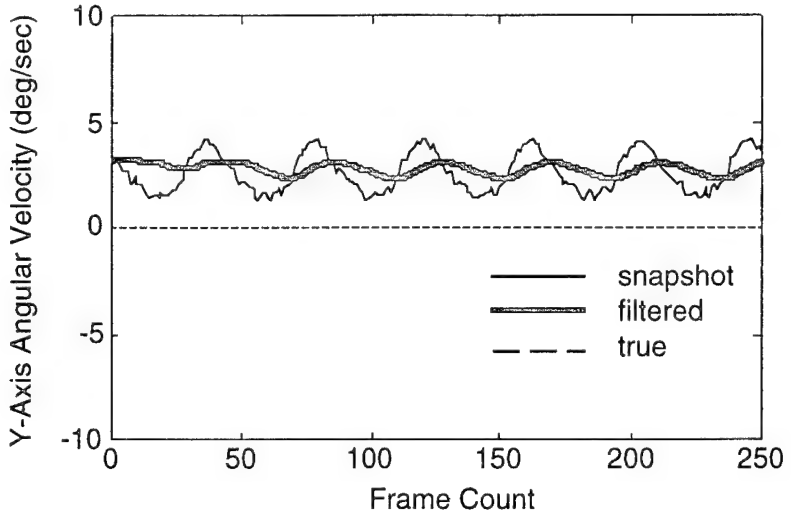


Figure 4.2-6: Pitch Angular Velocity Estimation History

Figures 4.2-7 to 4.2-9 illustrate the snapshot and filtered estimation error histories for the motion state and terrain parameters. As shown in figure 4.2-7, the snapshot impact time estimate has an oscillating error of 0% to 15%, which is reduced to within a 5% error zone after Kalman filtering.

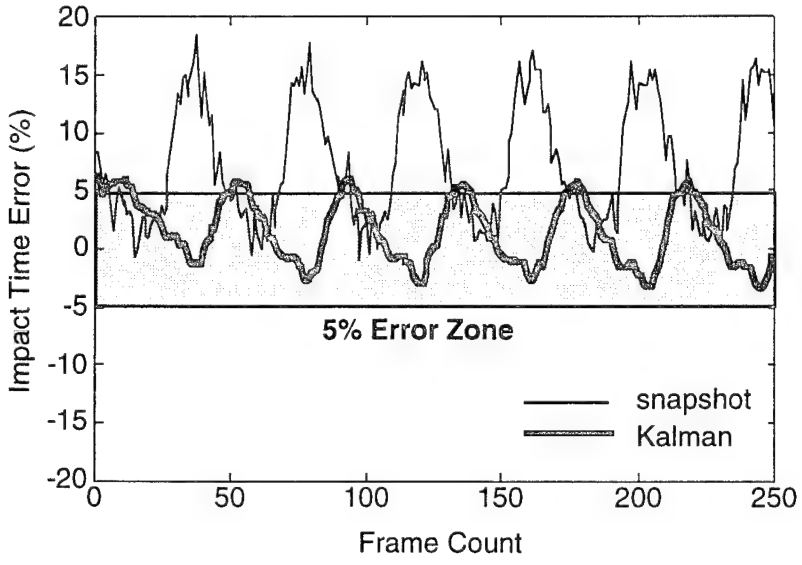


Figure 4.2-7: Impact Time Error History

Figure 4.2-8 illustrates the snapshot and the filtered aimpoint estimation error histories. The

snapshot estimation error oscillates between 2° to 4°, while the filtered estimation error converges to about 1.2°.

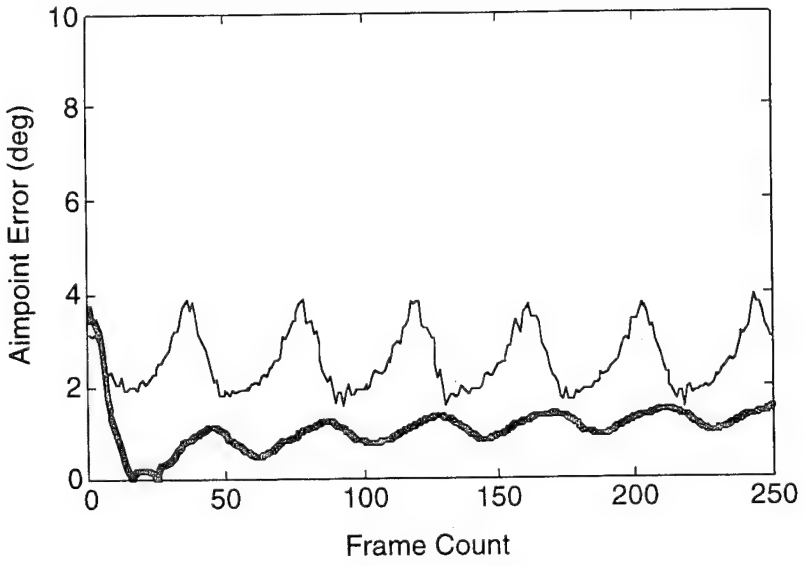


Figure 4.2-8: Aimpoint Error History

Figure 4.2-9 illustrates the snapshot and the filtered angular rate estimation error histories. The snapshot estimation error shows an oscillating error between 1.5 to 4°, while the filtered estimation error shows a slightly oscillating 3° error.

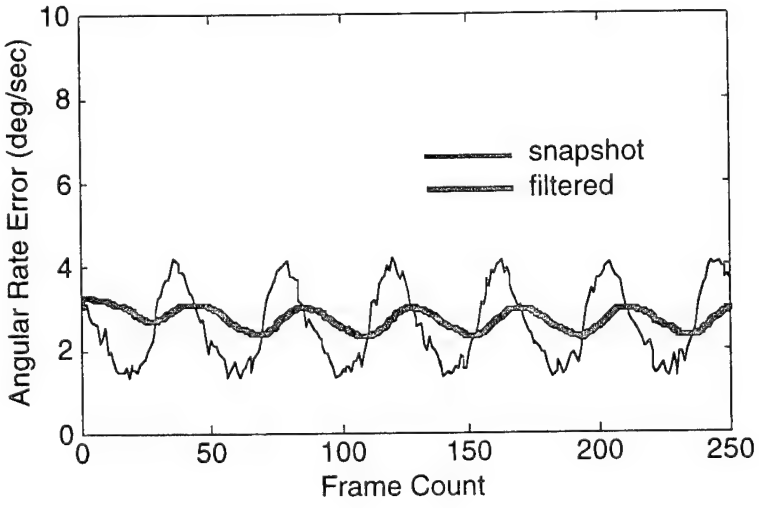


Figure 4.2-9: Angular Rate Error History

RMS errors, using the geometry based estimator, in the filtered impact time, aimpoint, and angular rate are 3.02%, 1.09°, and 2.77°/sec, respectively.

4.2.2 Structure Based Estimator

Figures 4.2-10 to 4.2-16 illustrate the motion state and terrain parameter estimation time histories for ATAS using the structure based estimator. The system used a structure of 100 layers with a 0.01 equal relative distance between layers. It also assumed a system model standard deviation 0.01° for heading and path angles, 0.01 °/sec for each element of the angular velocity vector. The discount factor for terrain impact time (range) map evolution was set at 0.9.

In the figures, the solid line represents the snapshot estimates, the thick solid line represents the Kalman filter smoothed estimates, and the dashed line represents the actual observer state and terrain shape parameters.

Figure 4.2-10 shows time histories of the estimated impact times. As shown, both the snapshot and Kalman filtered impact times oscillate around the actual impact time. Note that a larger variation occurs in the snapshot estimate, and the effects of the Kalman filtering in smoothing the oscillation of the snapshot estimate and improving estimation accuracy are clearly demonstrated.

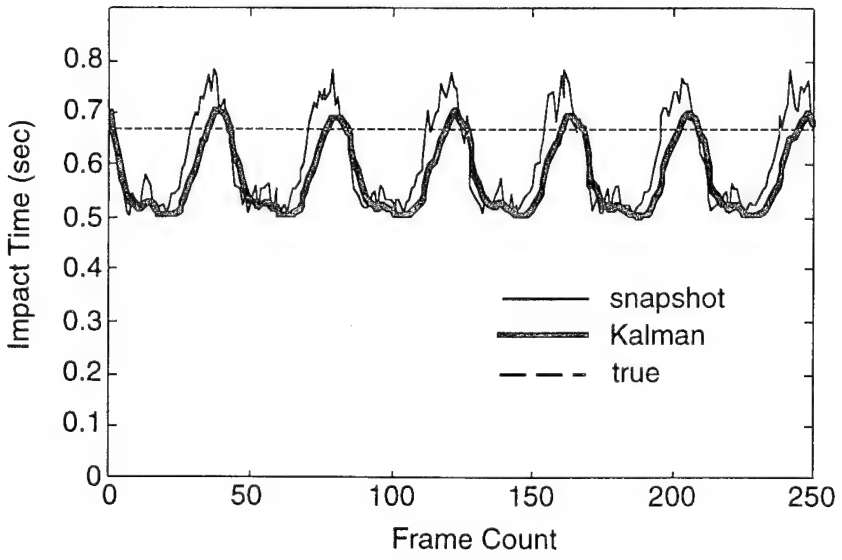


Figure 4.2-10: Impact Time Estimation History

Figures 4.2-11 and 4.2-12 illustrate time histories of the estimated heading and flight path angles, respectively. As illustrated, the snapshot heading angle estimate oscillates within a 1.0° range around the true heading vector, while the snapshot flight path angle estimate oscillates in a ±5° range. The filtered heading vector converges to the true heading angle, while the filtered flight path angle converges with about a 2.5° bias.

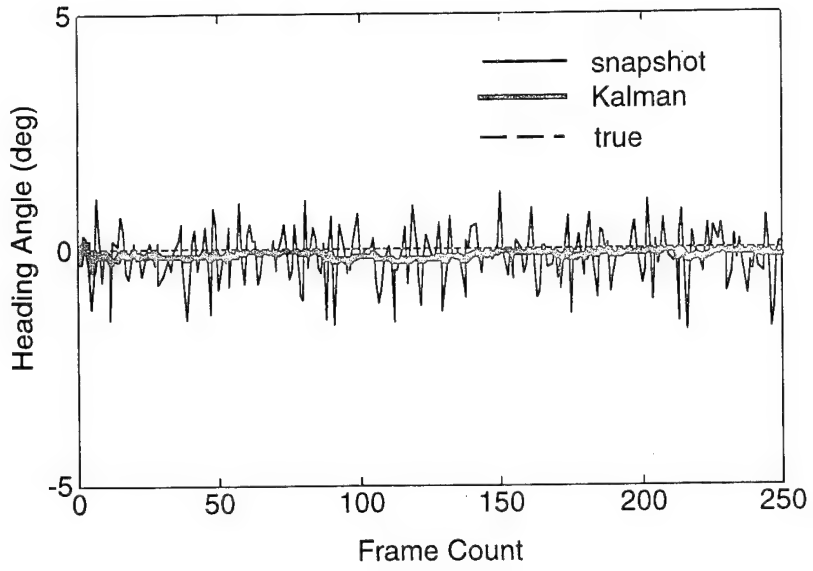


Figure 4.2-11: Heading Angle Estimation History

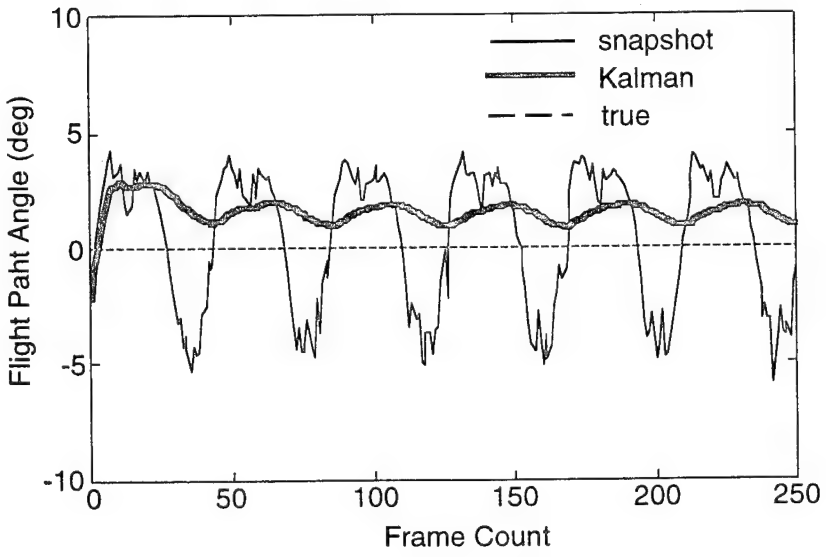


Figure 4.2-12: Flight Path Angle Estimation History

Figure 4.2-13 shows a time history of the estimated pitch (y-axis) angular velocity. As shown, the snapshot estimate oscillates about the true angular velocity, while the filtered estimate oscillates with a bias. The x and y-axis angular velocity estimates are not shown. They have a small error of less than 0.01°.

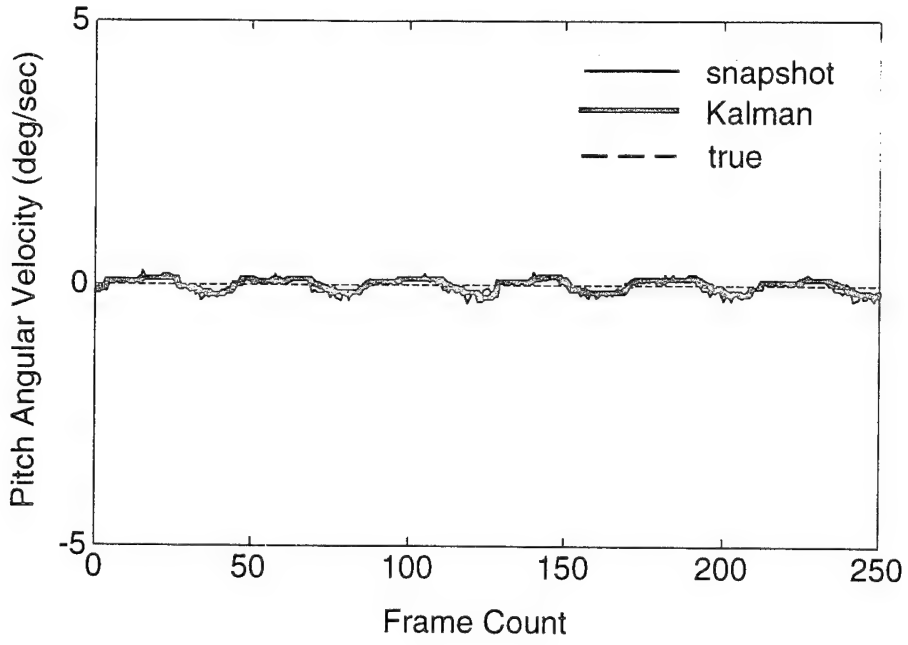


Figure 4.2-13: Pitch Angular Velocity Estimation History

Figures 4.2-14 to 4.2-16 illustrate the filtered estimation error histories for the motion state and terrain parameters. As shown in figure 4.2-14, the filtered impact time error oscillates around a negative 10% bias toward the smaller impact time.

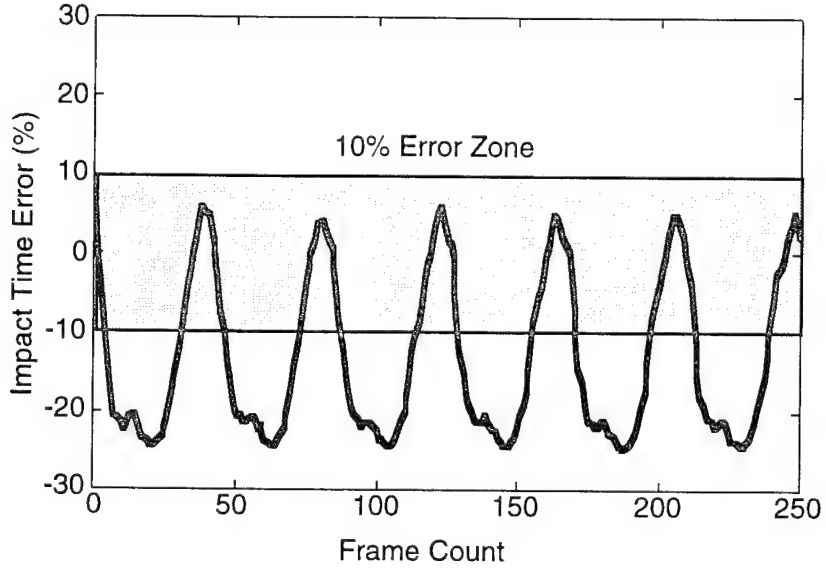


Figure 4.2-14: Impact Time Error History

Figure 4.2-15 illustrates the filtered aimpoint estimation error history. The filtered estimation error oscillates between 1° to 2°.

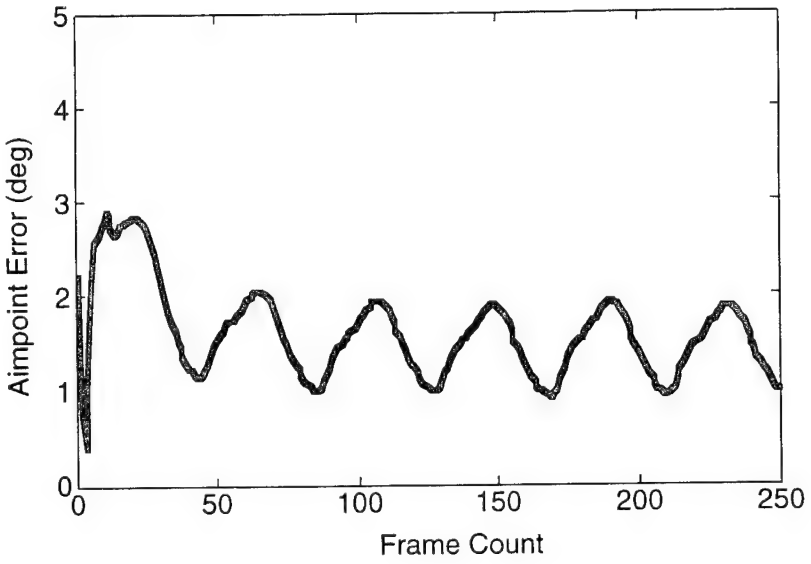


Figure 4.2-15: Aimpoint Error History

Figure 4.2-16 illustrates the filtered angular rate estimation error history. The filtered angular velocity estimation has a very small error of less 0.2°.

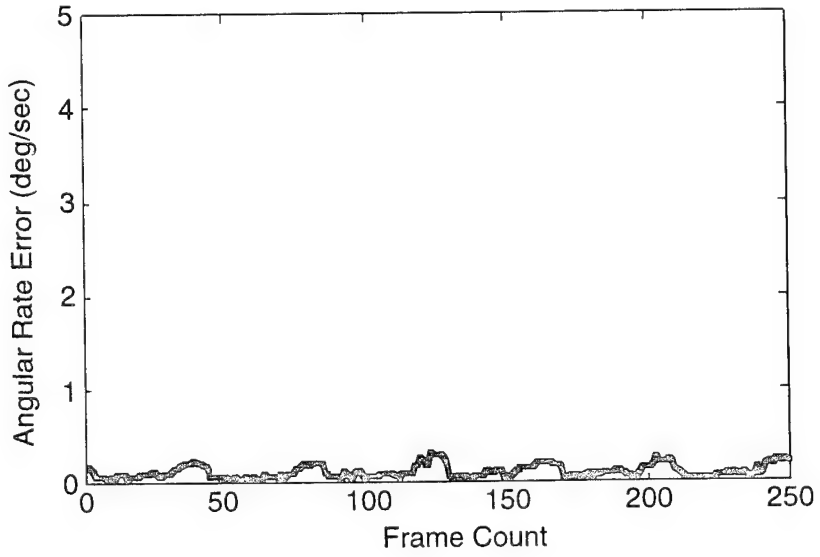


Figure 4.2-16: Angular Rate Error History

RMS errors, using the structure based estimator, in the filtered impact time, aimpoint, and angular rate are 17.02%, 1.52°, and 0.13°/sec, respectively.

4.2.2 Comparison of ATAS Predictions with Human Experiments

We now compare results generated by ATAS with some results obtained in two experiments

conducted by Warren (1976) and Grunwald & Kohn (1993). Detailed explanations of the experiments are given in the respective references. Section 4.2.2.1 presents the comparison with Warren's experiment, while section 4.2.2.2 presents the comparison with Grunwald and Kohn's experiment.

4.2.2.1 Comparison with Warren's Experiment

In the experiment conducted by Warren (1976), subjects were "flown" over a flat terrain decorated by a uniformly random array of luminous dots. The flight path was straight and level, with an altitude of d units and speed 1.25d units/s, where d was the average inter-dot spacing of the terrain texture. The lateral FOV was 53.1 deg, and the vertical FOV was 26.5 deg below the displayed horizon. After viewing the display, the subjects were asked to indicate their lateral heading aim point, and were able to do so with an accuracy of 1.5 deg standard deviation (across subjects).

To simulate the experimental environment, a sequence of computer images were generated by using the ATAS flat terrain image generator described in section 3.1 The simulated vehicle flew straight-and-level at an altitude of 100 feet above the terrain, with zero body attitude rates. The frame time between images was set as 0.033 s (30 Hz) for consistency with our analysis of the Grunwald & Kohn (1993) experiment (see below). The terrain scene was approximated by a two-dimensional sum-of-sines spatial texture pattern, having a fundamental wavelength of 100 ft, with coefficients chosen to approximate the luminous dot pattern used in the original experiment. The image plane was set normal to the observer's velocity vector with a zero depression angle relative to the terrain. The sensor FOV was set to 53.1° laterally by 53.1° vertically, and the pixel count was set at 100 laterally by 100 vertically, for a total of 10,000 pixels.

Figure 4.2-17 shows one image frame generated under the above conditions. The upper half of the figure is sky, and has a neutral gray level. The lower portion below the horizon shows the array of luminous dots, which naturally recedes to the horizon. The effects of pixellation are clearly evident.

In the simulation, a sequence of these images are generated, and then processed, frame by frame, using the previously described algorithm to obtain estimates of the observer state and terrain parameters. Figure 4.2-18 illustrates a typical resulting time history of the heading error, computed as the difference between the actual and estimated heading angle. The thin line shows estimation error generated by the *snapshot* least squares estimator. The high frequency content of this signal directly reflects the frame-by-frame variation in the image. The thick line shows estimation error generated by the Kalman filter, which provides smoothing by incorporating the observer dynamics. As can be seen, the algorithm provides excellent estimation of the observer's

heading by effectively exploiting the available temporal and spatial visual information.

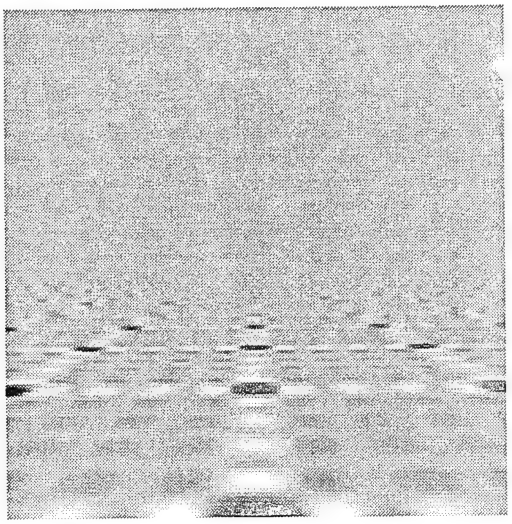


Figure 4.2-17: CGI Used for Simulating Warren Experiment

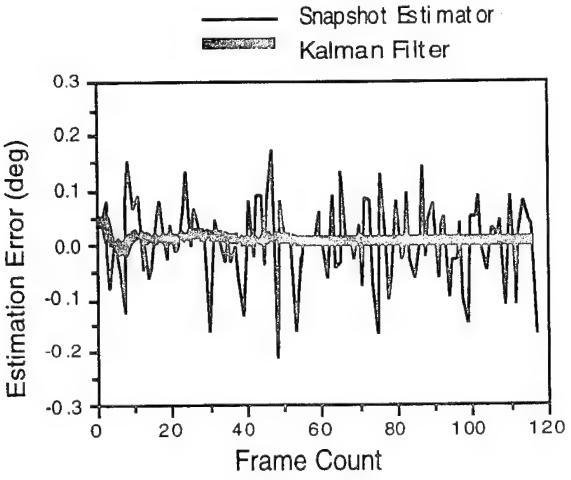


Figure 4.2-18: Heading Estimation Error Time History (Zero Image Noise)

The level of performance clearly exceeds that achieved by human observers but this is to be expected given the essentially error-free optimization implemented by the model. To better match human empirical data, we assume that the human visual system is not noise free, but can be modeled as a noisy generator of the ideal image. In our simulation, we implemented this by adding zero mean Gaussian random noise directly to the (pixellated) image, choosing the standard deviation (SD) of the image noise level to be fixed percentage of the overall gray level of the image. Figure 4.2-19 illustrates the resulting heading error estimation time history with a 10% image noise level (noise SD equals 10% of the overall image intensities). Clearly, this injection of image noise significantly increases the magnitude of the *snapshot* or least squares

estimate (note scale differences with figure 4.2-18), as well as that of the Kalman filter estimate.

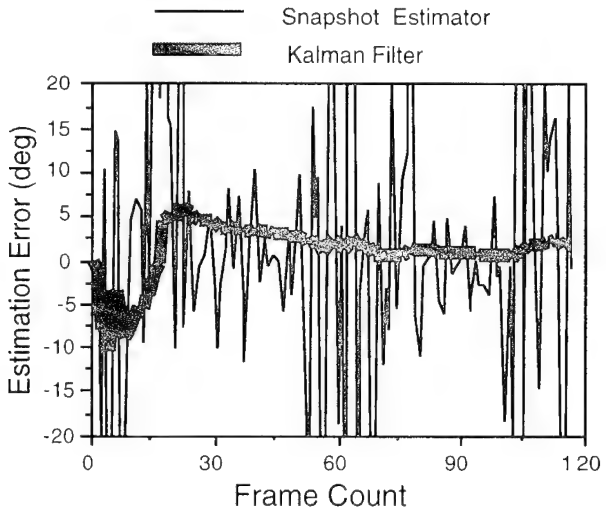


Figure 4.2-19: Heading Estimation Error Time History (10% Image Noise)

With this facility for injection of image noise, we then conducted a number of Monte Carlo simulations to generate statistics of the resulting estimation error, for different image noise levels. For each run, we computed the root mean squared (RMS) estimation error over a response time window, starting from 1.5 sec following stimulus presentation (assuming subjects needed at least 1.5 sec to respond and running to 4.0 sec following stimulus presentation (when model response suggests no further reduction in the estimation error). We then computed the mean and standard deviation of the RMS estimation error across 30 Monte Carlo simulations of the same experimental condition. This process was then replicated for several different image noise levels.

Figure 4.2-20 shows mean (triangles) and standard deviation (error bars) of the RMS estimation error as predicted by the model for three different image noise levels: 5%, 10%, and 20% (the curve is an interpolation between conditions). We see monotonically increasing error with image noise level. We have also indicated on the figure the 1.5 deg estimation error obtained under the experiment conducted by Warren (1976), and have indicated how this performance level can be "explained" as an 8% perceptual image noise level, within the context of the model we have presented here.

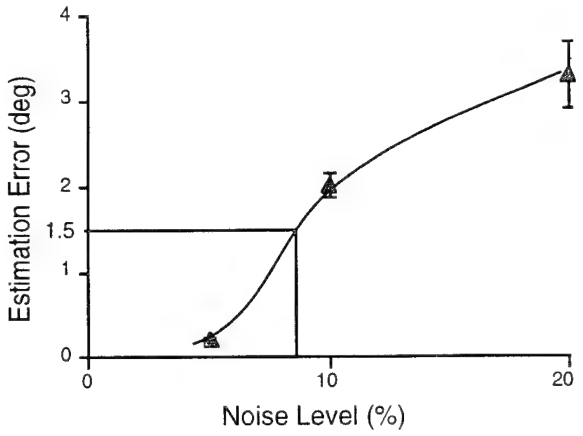


Figure 4.2-20: Estimation Error vs. Perceptual Noise Level

4.2.2.2 Comparison with Grunwald and Kohn's Experiment

In one portion of the experiment conducted by Grunwald and Kohn, a subject was "flown" over flat terrain decorated by a field of rectangular patches placed in a cross-grid pattern. As with Warren's experiment, the flight path was straight and level and the image plane was normal to the observer's velocity vector. The field of view was 102.7 deg horizontal and 90 deg vertical. Observer altitude was set at d units above the ground plane. The terrain was visible from a distance 1d until 15d units from the observer. The average length and width of the viewed rectangles was 0.385d, and the average distance between their centers was 0.625d. In each trial, the lateral flight path deviated from the viewing axis by the sideslip angle β, which was chosen from a uniformly distributed random set ranging from -45 to +45 deg. The subject's task was to place a marker on the estimated aimpoint. The subject was given a maximum of 8 sec to respond, after which the run was terminated. In the experiment, different velocity-to-height ratios were used, ranging from 0.25 to $4s^{-1}$.

In our simulation of this experiment, the simulated vehicle also flies straight-and-level at an altitude of 100 feet above the terrain, with zero body attitude rates. The frame time between images was set as 0.033 s (30 Hz), which is the screen update frequency of the Silicon Graphics IRIS 4D 50/GT workstation used in the experiment. The terrain scene was approximated by a two-dimensional sum-of-sines texture pattern, of regularly distributed square patches of size 38.5 ft, spaced 77.0 ft apart. The image plane was set normal to the observer's velocity vector. The sensor FOV was set to 90° laterally by 90° vertically, and the pixel count was set at 100 laterally by 100 vertically, for a total of 10,000 pixels.

Figure 4.2-21 shows one image frame generated under the above conditions. Figure 4.2-22 illustrates a typical time history of the heading error, obtained during our Monte Carlo analysis

of this experiment, using a 15% image noise level to simulate human perceptual limitations. As before, the thin line shows the output of the snapshot least squares estimator (before filtering) and the thick line shows the output of the Kalman filter.

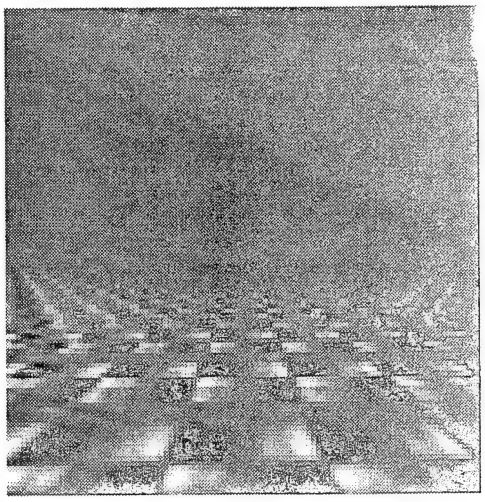


Figure 4.2-21: CGI used for Simulating Grunwald and Kohn's Experiment

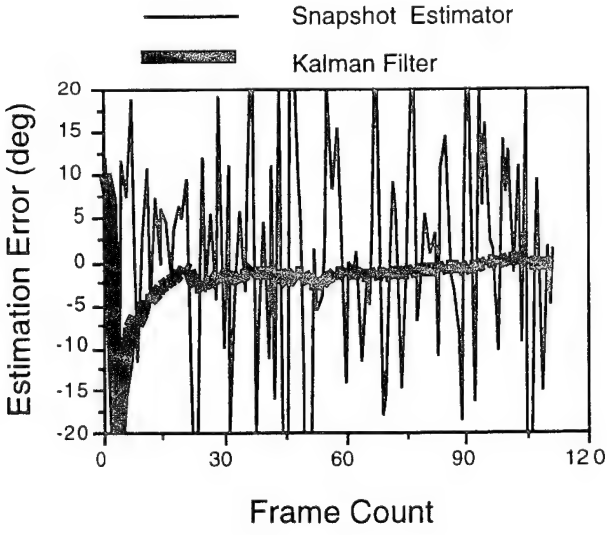


Figure 4.2-22: Heading Estimation Error Time History (15% Image Noise)

As with our analysis of the experiment by Warren, we varied the image noise level to find a best match to the experimental data, across all conditions tested. For this experiment, we found a 15% image noise level to provide the best overall model match to the data. Figure 4.2-23 shows the resulting RMS estimation error in aimpoint, as a function of velocity-to-height ratio (the "global flow rate" of Warren). Model predictions are indicated by the shaded region, defined as the plus/minus one-sigma envelope about the expected RMS error. Experimental data points are

indicated by the solid circles. Note that the data shows a clear trend in which errors vary inversely with velocity-to-height ratio, with the lower flow rates yielding higher errors. This trend is replicated by the model results, as we expect with extreme (in this case, very low) flow rates. Although not covered in this experiment, we would expect a gradual increase in aimpoint estimation error, as flow rates are increased, so that an overall U-shaped function is obtained.

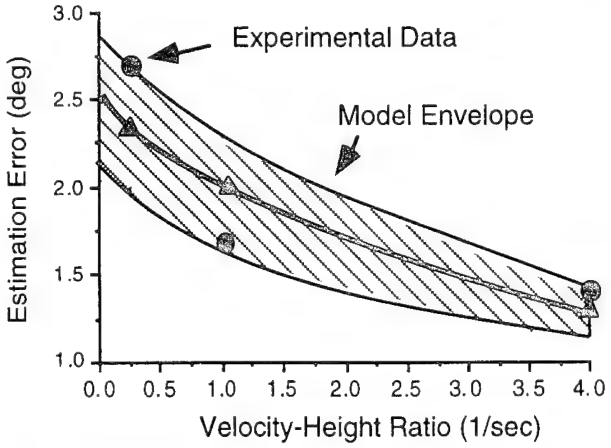


Figure 4.2-23: Heading Estimation Error vs. Velocity Height Ratio (Data vs. Model)

Using ATAS, we analyzed several earlier experimental studies of human egomotion perception. For the two reported on here, we used simple additive image noise to model the human's inability to register perfectly noise-free images, and we were able to show how this "perceptual noise" leads to corresponding inaccuracies in estimates of egomotion state. In particular, we showed how, within the model context, an assumed 10% to 15% image noise level leads directly to model-predicted egomotion aimpoint accuracy in the experimentally observed range of 1.5 to 2.5 deg. In addition, we showed how these aimpoint accuracy can be expected to co-vary with velocity-to-height ratios, which change by over a factor of 10, and confirmed these trends with experimental data showing the same trends. Although more model validation is clearly needed, we feel that these initial matches between model and data are very encouraging.

4.3 Evaluation Using Non-Flat Terrain CGI

We also tested the egomotion and terrain shape estimator on complicated non-flat terrain images, specifically, the Yosemite sequence, which was created by Lynn Quam and provided to us by Dr. J. L. Barron (1994). Figure 4.3-1 shows one frame of the 15 frame Yosemite sequence.

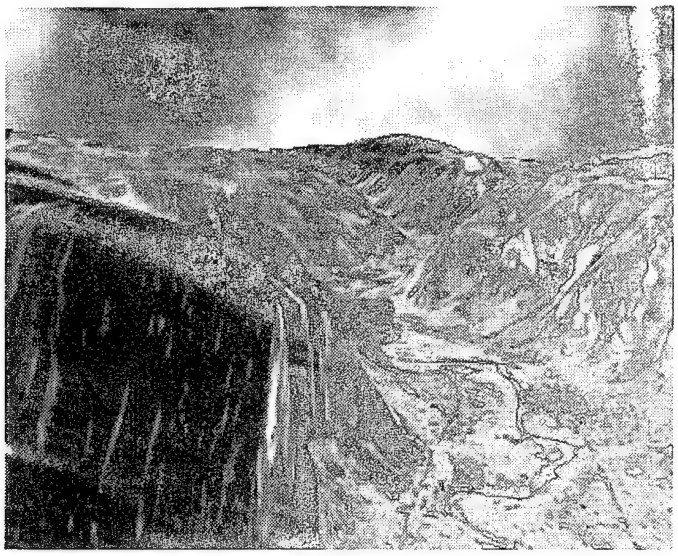


Figure 4.3-1: Single Frame of Yosemite Image Sequence

This sequence is challenging because of its complicated terrain structure, which is beyond the capability of any simple polynomial geometry model (flat, quadratic, etc.). As a matter of fact, when we tried to use the geometry-based estimator for egomotion and terrain shape estimation, the results were not even remotely correlated with the true motion state and terrain shape.

We used the structure-based estimator for egomotion and terrain shape estimation of the Yosemite sequence. For the snapshot estimator, a 512 equal distance layer structure was assumed with the relative distance between layers set as 0.01. For the Kalman filter, the model standard deviations for the heading and path angles were set at 0.01°, and the model standard deviations for the angular velocity vector were set at 0.01 °/sec for each individual vector element. The discount factor for terrain impact time (range) map evolution was set at 0.9.

Figures 4.3-2 and 4.3-3 illustrate the intensity-coded third and ninth depth (impact time) maps generated directly from the estimator without any post run image processing (filtering, sharpening, noise reduction, etc.). Lighter pixels in the map are closer to the observer. The white pixels are either pixels that are above the horizon or pixels that have zero image gradients, thus have no information available for structure extraction.

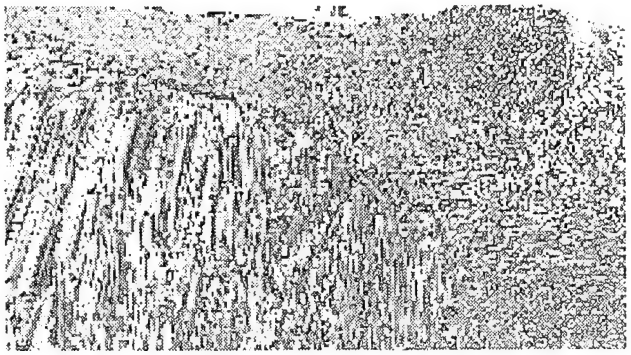


Figure 4.3-2: The Third Impact Time Map Generated from Yosemite Sequence

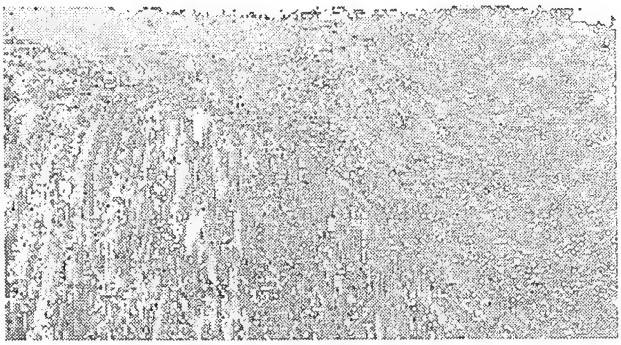


Figure 4.3-3: The Ninth Impact Time Map Generated from Yosemite Sequence

Note that the third map has many white pixels, indicating unknown depths (impact times) at those points due to lack of gradient information (or image flow). After six frames of Kalman filter integration of the depth maps, the ninth map shows a much smoother impact time map with fewer white pixels. Although no quantitative comparison between the true and estimated impact time map is provided, qualitatively, it is evident that the structure-based estimator has a capability to recover the structure of the terrain.

For egomotion state estimation, the actual heading vector was given as (0.0, 0.17, 0.98) by Heeger and Jepson (1992) and was constant over the 15 frame sequence. The estimated heading vector was (-0.10, 0.18, 0.98) for the first estimate (the third frame) and (-0.09, 0.18, 0.98) for the last estimate. Figure 4.3-4 shows the aimpoint error history for the 15 frames. As shown in

the figure, the average aimpoint error was less then 5° over the 15 frame sequence.

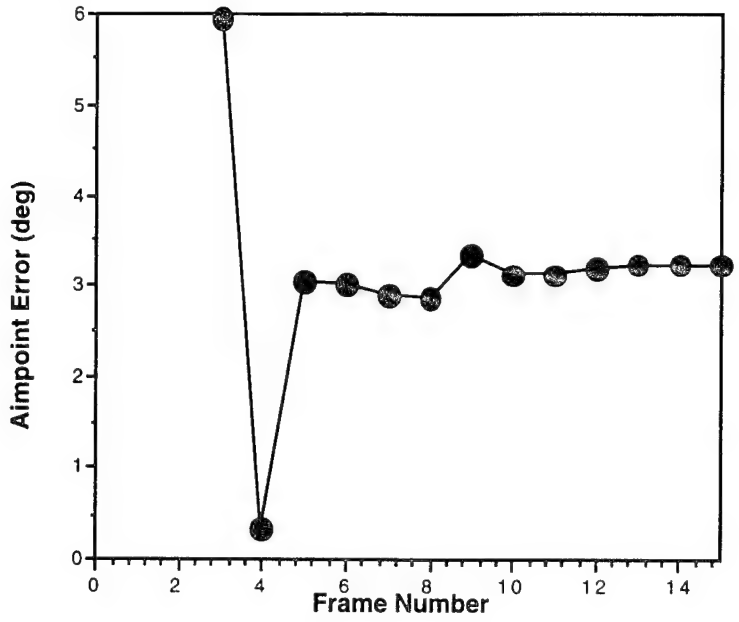


Figure 4.3-4: Aimpoint Error History

The actual rotation rate was given as 0.095 degrees/frame. Figure 4.3-5 shows the angular rate error history for the 15 frames. The estimated average rotation rate over the 15 frame sequence was 0.032 °/frame. The angular rate error was less than 0.06° over the 15 frame sequence.

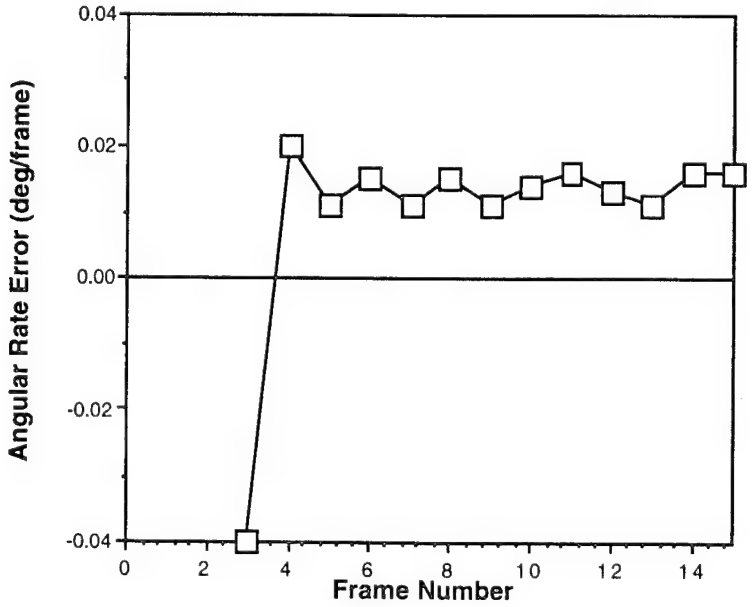


Figure 4.3-5: Angular Rate Error History

4.4. Evaluation Using Real Imagery

Evaluation using real imagery was done using the flight-recorded and real-time frame grabbed terrain image sequence obtained from the NASA Ames Research Center. The image sequence was acquired by a camera mounted under a helicopter nose and oriented roughly in the direction of flight. The position of the camera and its orientation with respect to the helicopter were held constant throughout the flight. The methodology used to generate the image sequence and truth data are described in Smith (1990). The helicopter was flying with a speed of approximately 20 knots at an altitude of 15 ft above a runway. The images were captured at 30 Hz, so that the helicopter moved approximately 1 ft between successive images. Figures 4.4-1a and 4.4-2b show the first and the last images of a sequence of 90 images. Each image has a resolution of 442 x 512 pixels at 256 gray levels.

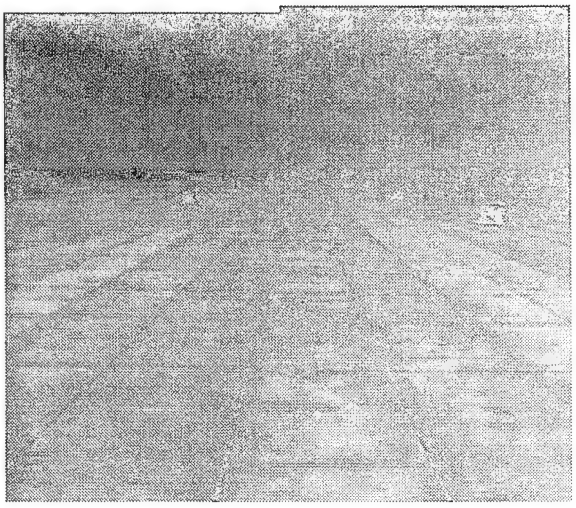


Figure 4.4-1a: First Image of NASA Sequence

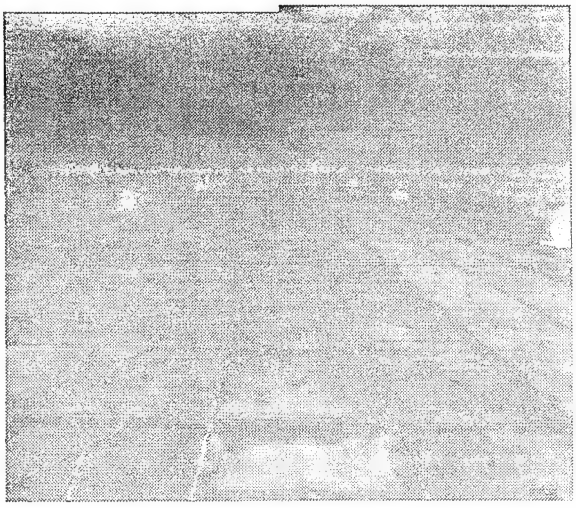


Figure 4.4-2b: Last Image of NASA Sequence

Using the same image sequence, we first evaluated the geometry based egomotion and terrain shape estimator (section 4.4.1), and then the structure based estimator (section 4.4.2).

4.4.1 Geometry-Based Estimator

The geometry model based estimator was tested first. For the snapshot estimator, a flat terrain model was adopted. For the Kalman filter, the model standard deviations for heading and flight path angle dynamics were set at 0.01⌋, amd the model standard deviations for angular velocity dynamics were set at 0.01 ⌋/sec^{2,} equivalently for three angular velocity components. The discount factor for terrain impact time (range) map evolution dynamics was set as 0.9. In processing the images, the pixels above the horizon were not used.

4.4.1.1 Testing Using the Original Images

We first used the images at their original 442 x 512 pixel resolution without any pre-processing of images. Figures 4.4-2 to 4.4-5 show time histories for the estimated helicopter impact time (altitude/speed), heading angle, flight path angle, and three individual angular velocity components. The plots are drawn by plotting the estimate and measured state and terrain parameters against frame count with a frame interval time of 1/30 sec. In the figures, the solid line represents the snapshot estimates, the thick solid line represents the Kalman filter smoothed estimates, and the dashed line represents the helicopter state and terrain parameters measured using on-board navigation systems.

As shown in figure 4.4-2, the impact time estimate has quite a large error. Two factors contribute to this. The first is the lack of intensity contrast in the "non-decorated" real imagery, as demonstrated in figure 4.4-1. The second is the small pixel size that leads to a high noise/signal ratio in the image gradient computation. These two factors, however, seem not affect the motion state estimation accuracy adversely. Figures 4.4-3 and 4.4-4 illustrate the time histories of the estimated heading and flight path angles, respectively. The ability of the system to track the helicopter motion state dynamics is clearly demonstrated in plots of the three angular velocity components, shown in figures 4.4-5, 4.4-6, and 4.4-7, respectively. Note how closely and swiftly each estimated angular velocity component follows the dynamics of its corresponding true angular velocity component.

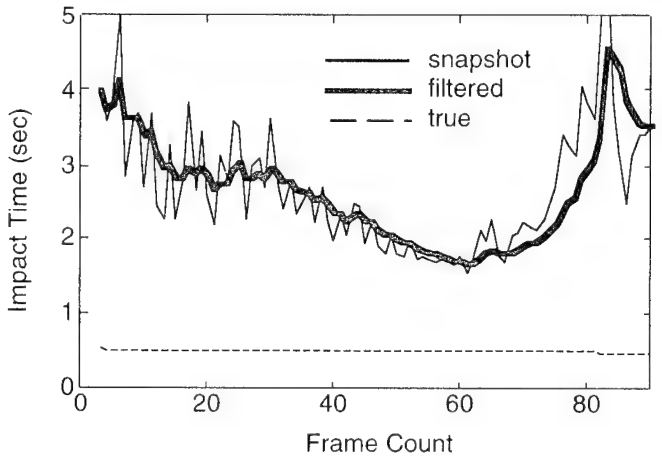


Figure 4.4-2: Impact Time Estimation History

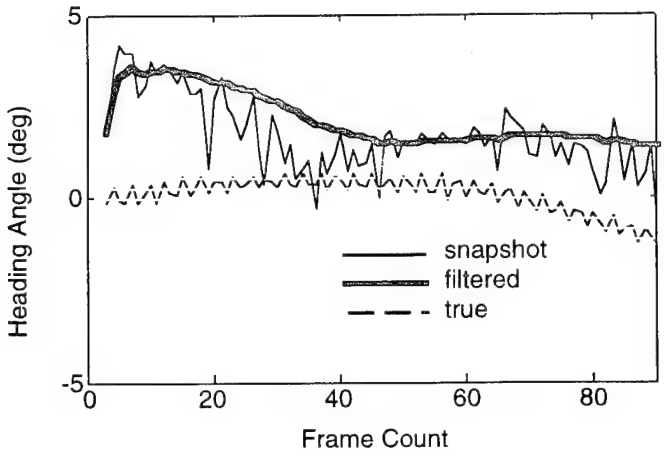


Figure 4.4-3: Heading Angle Estimation History

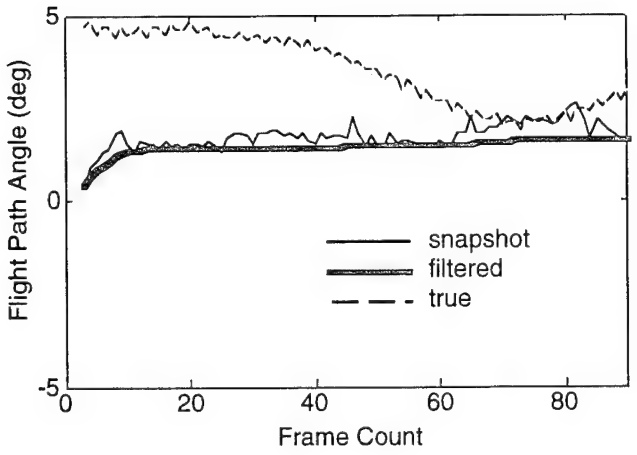


Figure 4.4-4: Flight Path Angle Estimation History

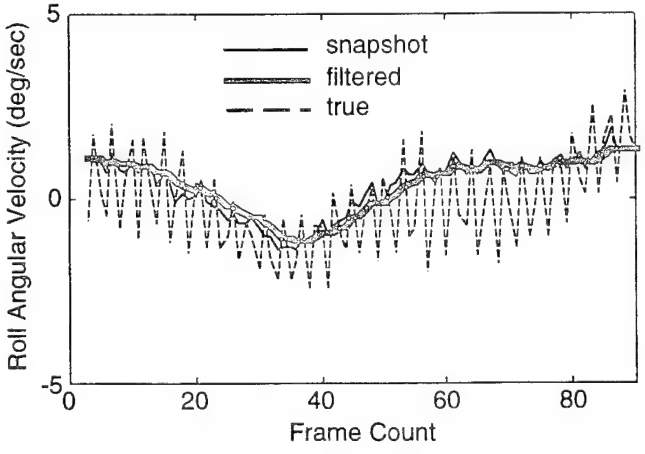


Figure 4.4-5a: Roll Angular Velocity Estimation History

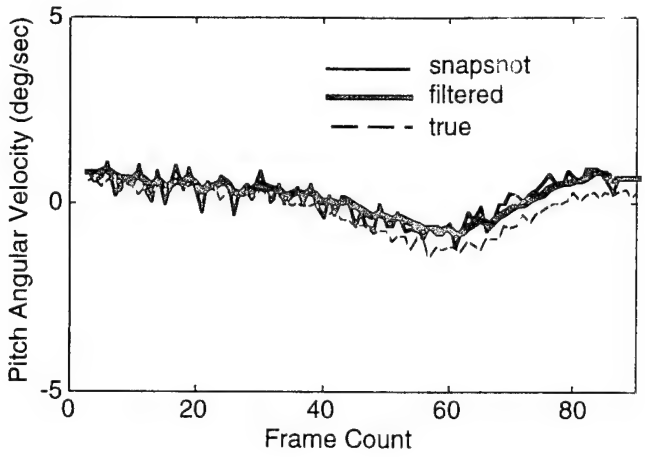


Figure 4.4-5b: Pitch Angular Velocity Estimation History

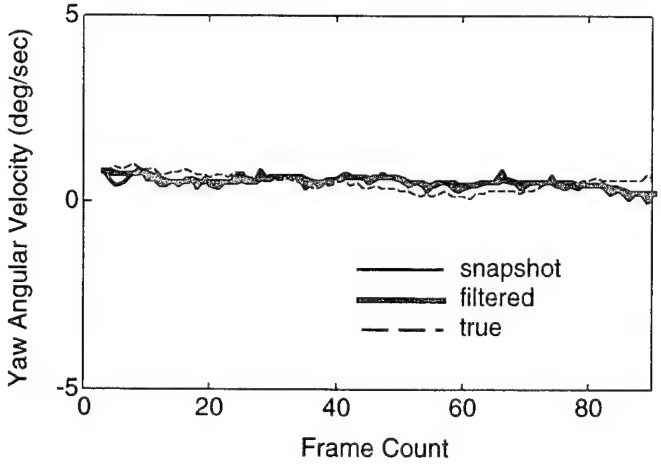


Figure 4.4-5c: Yaw Angular Velocity Estimation History

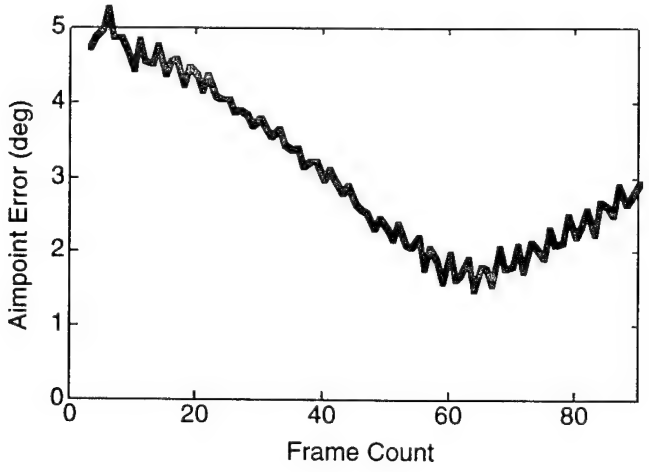


Figure 4.4-6: Aimpoint Error History

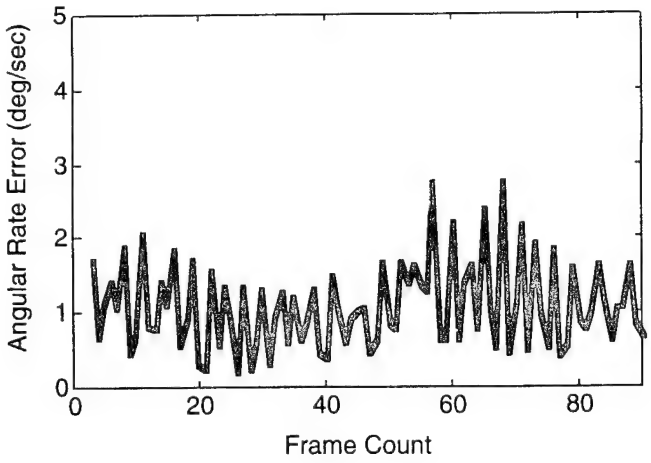


Figure 4.4-7: Angular Rate Error History

Figures 4.4-6 and 4.4-7 show time histories of the aimpoint error and angular rate error, respectively. After the first 30 frames (1 sec), the aimpoint error stays below 2.5°, and the angular rate error stays below 2°/sec. The RMS errors for aimpoint and angular rate are 2.40° and 1.29 °/sec, respectively.

4.4.1.2 Testing Using Superpixellated Images

Estimation using high resoluation images (442x512) leads to a rather large impact time errors. Since the model based estimator is best suited for simple geometry terrain without objects and uses only low-frequency terrain motion, superpixellation of the original images is called for. Figures 4.4-8a and 4.4-8b show the superpixellated versions of the images illustrated in 4.4-1a and 4.4-2b, at a resolution of 56 x 64 pixels, using 8 x 8 kernels. The trucks are no longer recognizable, but the low frequency imagery of the runway is.

Figure 4.4-8a: Superpixellated First Image of NASA Sequence

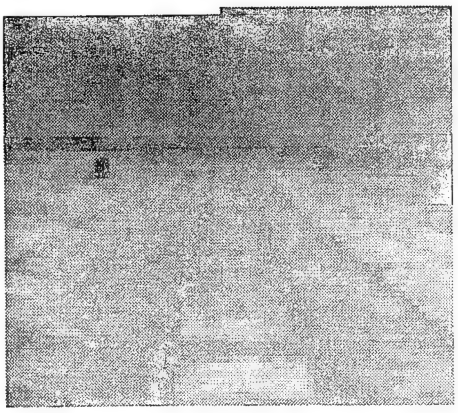


Figure 4.4-8b: Superpixellated Last Image of NASA Sequence

Figures 4.4-9 to 4.4-14 show the estimation time histories for the helicopter impact time (altitude/speed), heading angle, flight path angle, and three individual angular velocity components, respectively. As shown in figure 4.4-9, the impact time estimate now approaches the true impact time. The difference between the true and the filtered impact time is about 0.05 sec or 1.6 ft. This shows that for overall terrain surface estimation superpixellation does not adversely affect estimation accuracy but enhances it.

Figures 4.4-10a and 4.4-10b illustrate the time histories of the heading and flight path angle estimates, respectively. Superpixellation does not affect heading angle estimation; but seems to introduce a consistent negative flight path angle bias to the flight path estimate. Note again how the system estimates rapidly follow the trend of true helicopter motion states.

Angular velocity estimation is not adversely affected by superpixellation, as shown in figures 4.4-11a, b, and c. The three estimated angular velocity components track and follow the dynamics of the true angular velocity closely.

Figures 4.4-12, 4.4-13, and 4.4-14 show the time histories of the impact time error, aimpoint error and angular rate error, respectively. After the first 30 frames (1 sec), the impact time error almost stays within a 10% error zone, the aimpoint error stays below 3°, and the angular rate error stays below 2°/sec. The RMS errors for the impact time, aimpoint, and angular rate are 7.60%, 3.52°, and 1.93 °/sec, respectively.

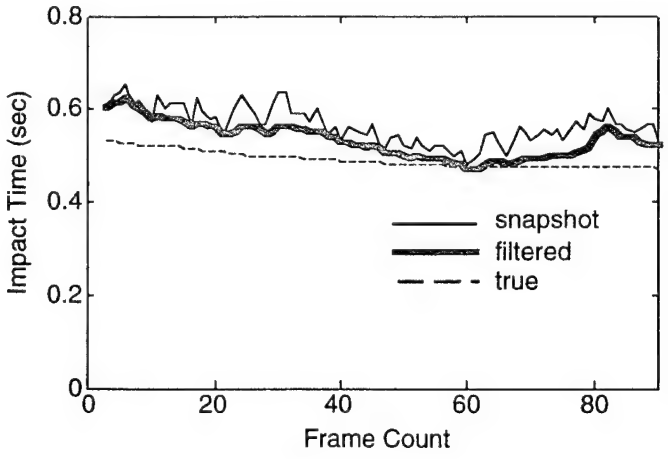


Figure 4.4-9: Impact Time Estimation History

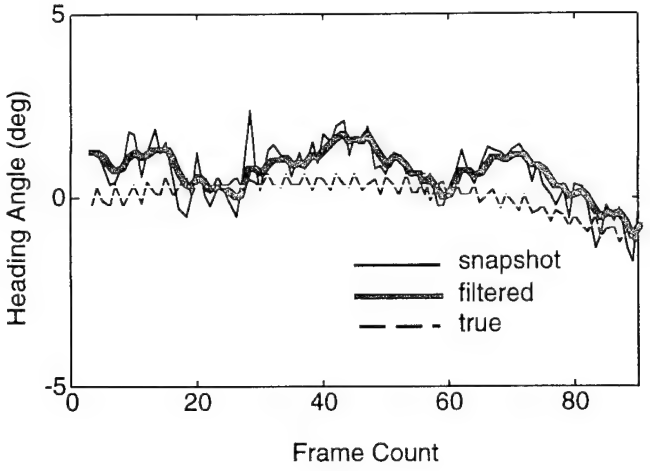


Figure 4.4-10a: Heading Angle Estimation History

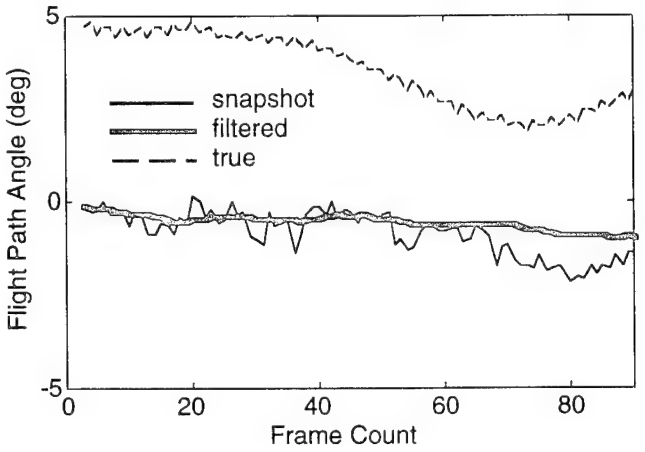


Figure 4.4-10b: Flight Path Angle Estimation History

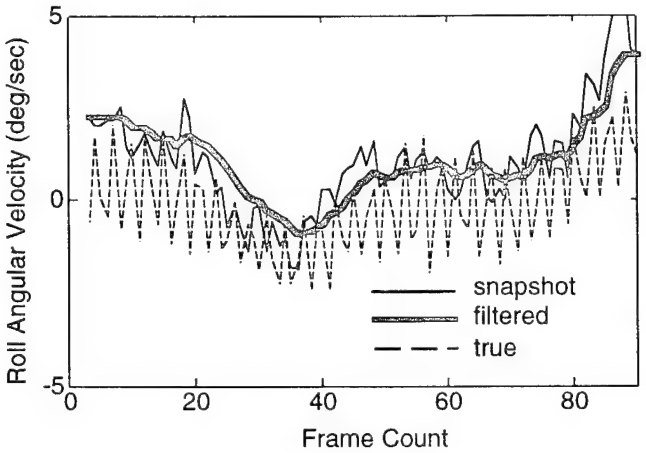


Figure 4.4-11a: Roll Angular Velocity Estimation History

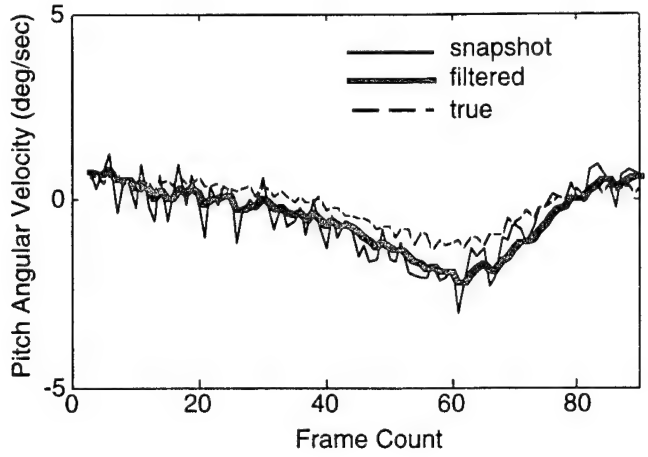


Figure 4.4-11b: Pitch Angular Velocity Estimation History

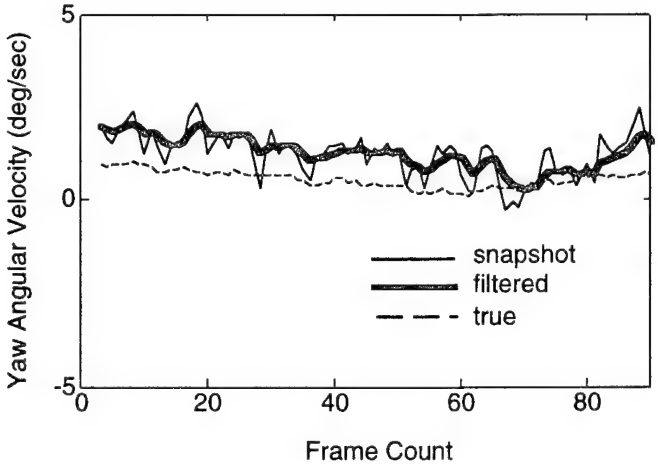


Figure 4.4-11c: Yaw Angular Velocity Estimation History

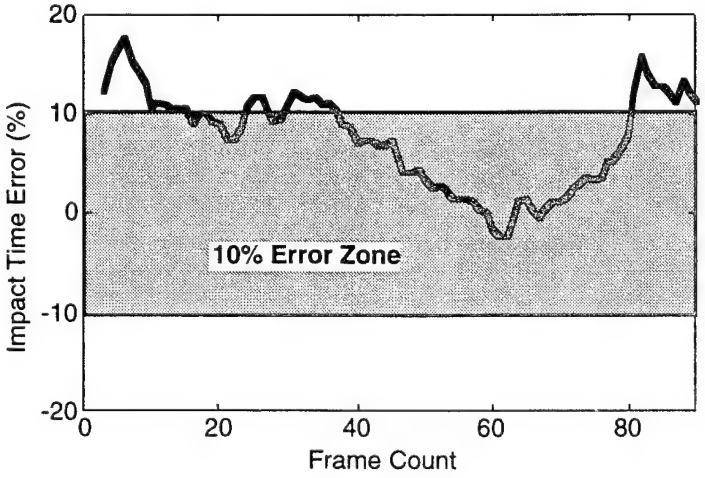


Figure 4.4-12: Impact Time Error History

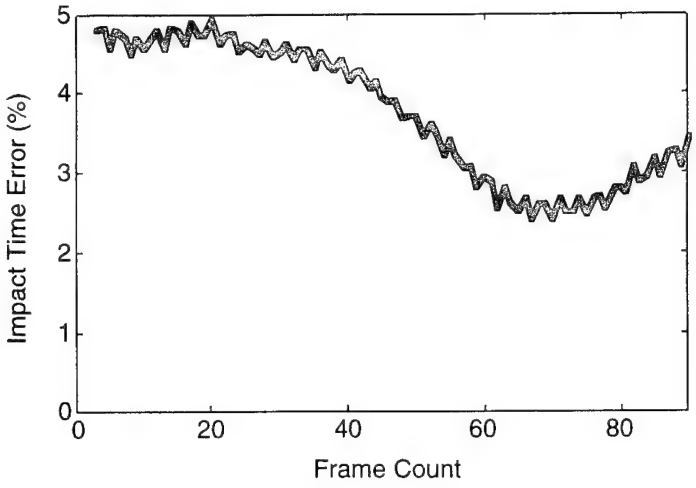


Figure 4.4-13: Aimpoint Error History

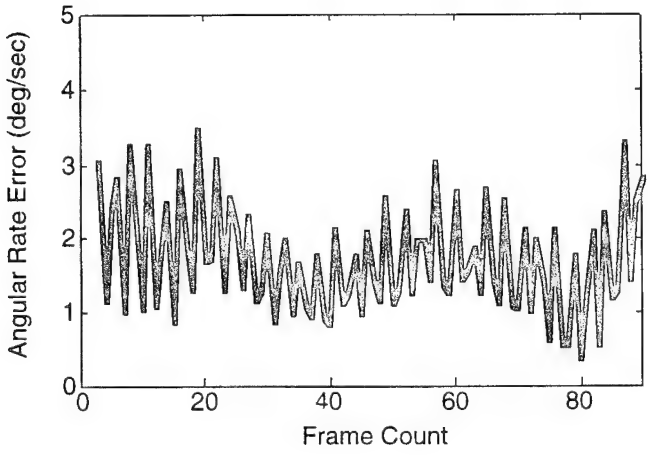


Figure 4.4-14: Angular Rate Error History

4.4.2 Structure-Based Estimator

For the structure-based estimator, a 256 layer structure was assumed. The relative distance between layers was set at 0.01. The Kalman filter system model standard deviations and the discount factor were the same as those used in the geometry-based estimator. Figure 4.4-15 to 4.4-20 show time histories for the estimated helicopter impact time, heading angle, flight path angle, and three individual angular velocity components.

We tested the estimator using both the original images and 8 x 8 kernel superpixllated images. Section 4.4.2.1 describes the results using the original images, while section 4.4.2.2 describes the results using the superpixellated images.

4.4.2.1 Testing Using the Original Images

Figure 4.4-15 shows a history of the estimated impact time. As shown, the impact time estimate has quite a large error. The same two factors, "non-decorated" real imagery and small pixel size, which were described in the geometry-based estimator, contribute to this large error. Note that the impact time error is smaller than that obtained using the geometry based estimator. Moreover, the structure based estimator was able to extract *more* than the terrain structure. Figures 4.4-16a, 4.4-16b, and 4.4-16c show the fifth, fifteenth, and twenty-fifth impact time maps obtained directly from the estimator. In the figures, blobs indicating the location of trucks are evident even without any post image processing.

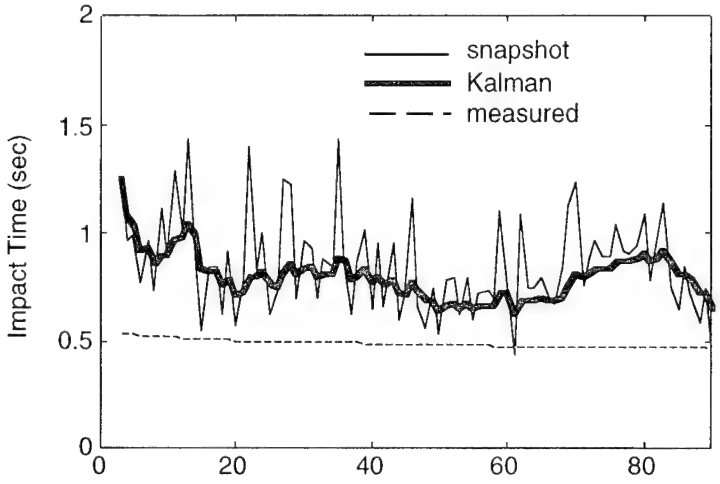


Figure 4.4-15: Impact Time Estimation History

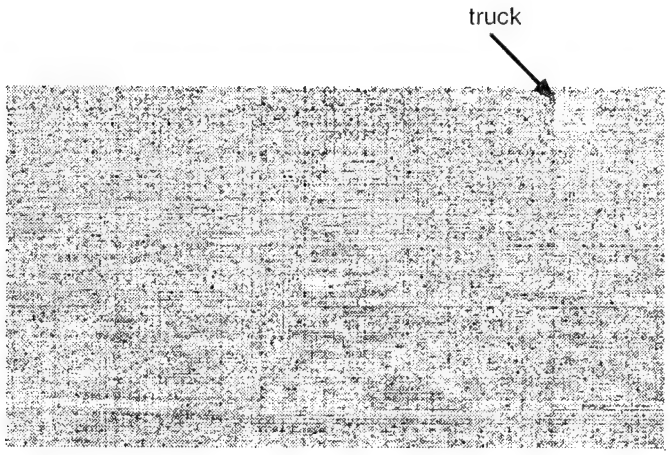


Figure 4.4-16a: The Fifth Impact Time Map

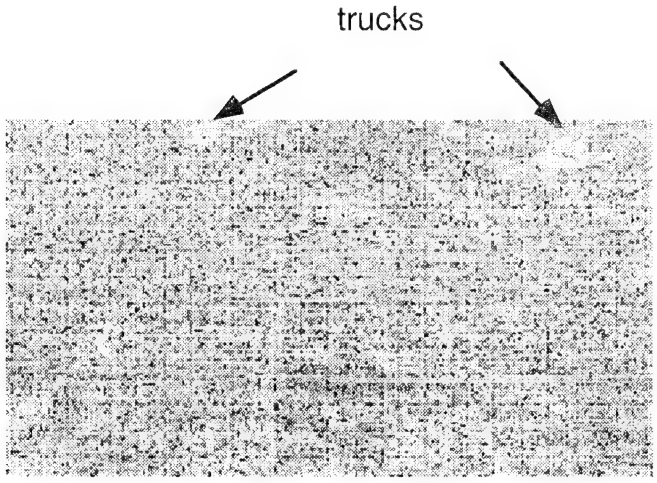


Figure 4.4-16b: The Fifteenth Impact Time Map

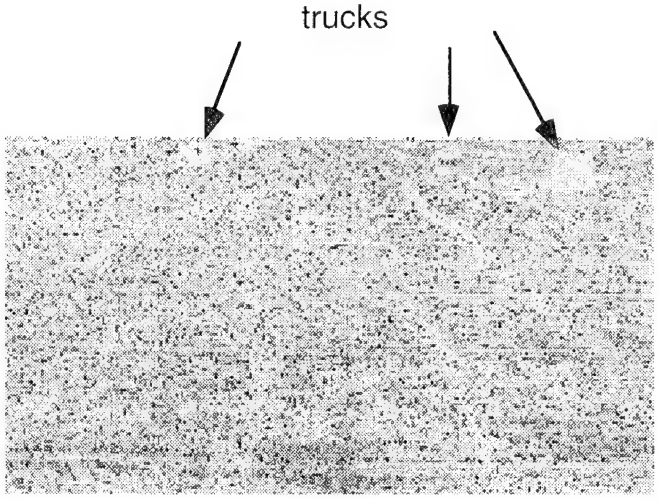


Figure 4.4-16c: The Twenty Fifth Impact Time Map

Figures 4.4-17a and 4.4-17b illustrate the time histories of the estimated heading and flight path angles, respectively. Figures 4.18a, 4.18b, and 4.18c show time histories of the three estimated angular velocity components, respectively. Note how the estimated motion states oscillate around the measured (true) motion states. The ability of the structure-based estimator to track the helicopter motion state dynamics is clearly demonstrated in these plots.

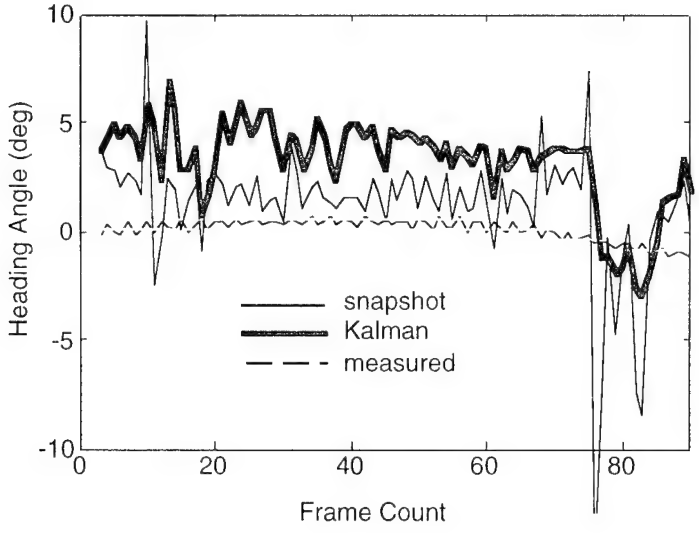


Figure 4.4-17a: Heading Angle Estimation History

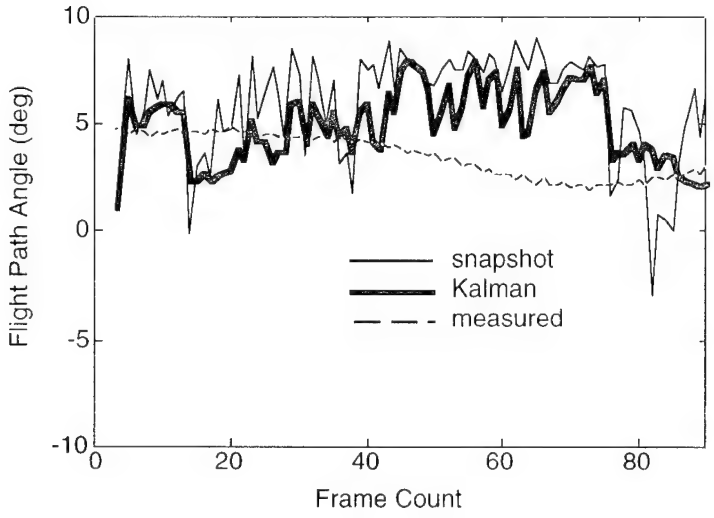


Figure 4.4-17b: Flight Path Angle Estimation History

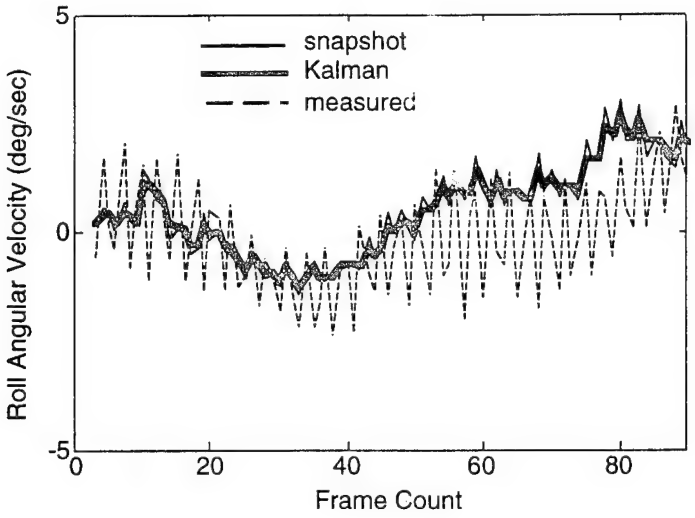


Figure 4.4-18a: Roll Angular Velocity Estimation History

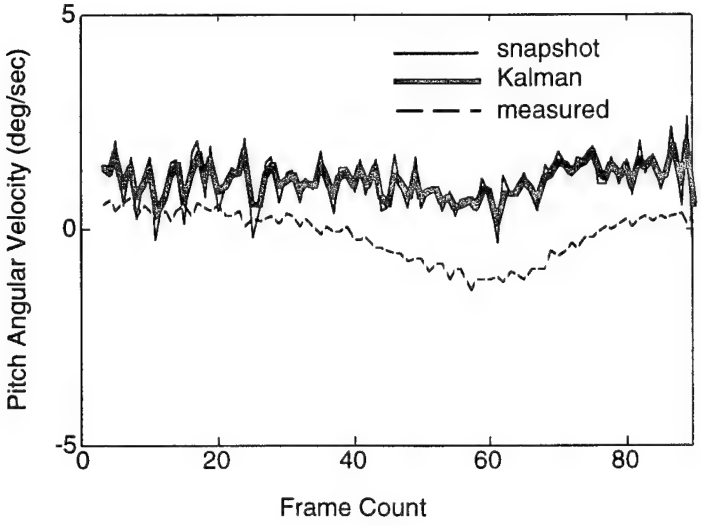


Figure 4.4-18b: Pitch Angular Velocity Estimation History

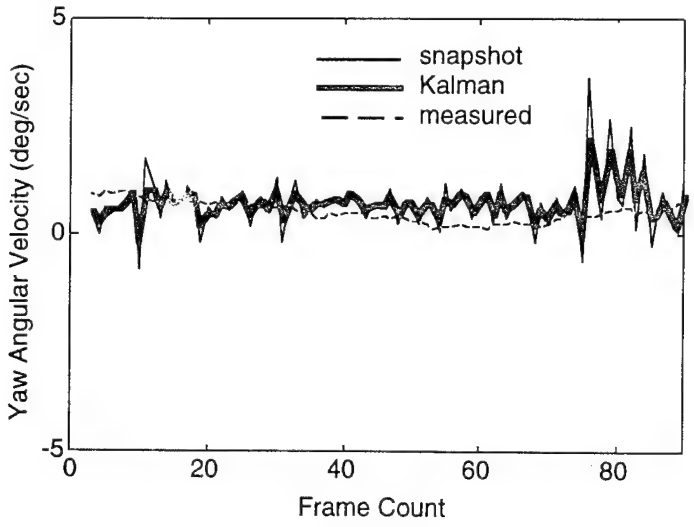


Figure 4.4-18c: Yaw Angular Velocity Estimation History

Figure 4.4-19a and 4.4-19b show time histories of the aimpoint error and angular rate error, respectively. The aimpoint error stays between 2° to 2.5°, while the angular rate error stays below 3.5°/sec.

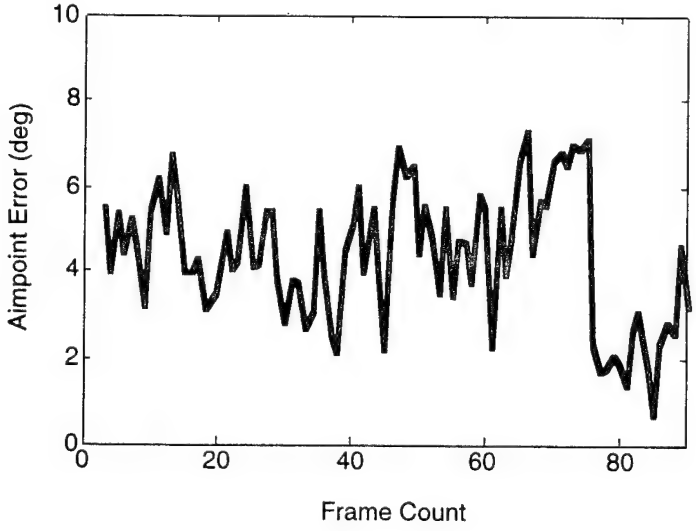


Figure 4.4-19a: Aimpoint Error History

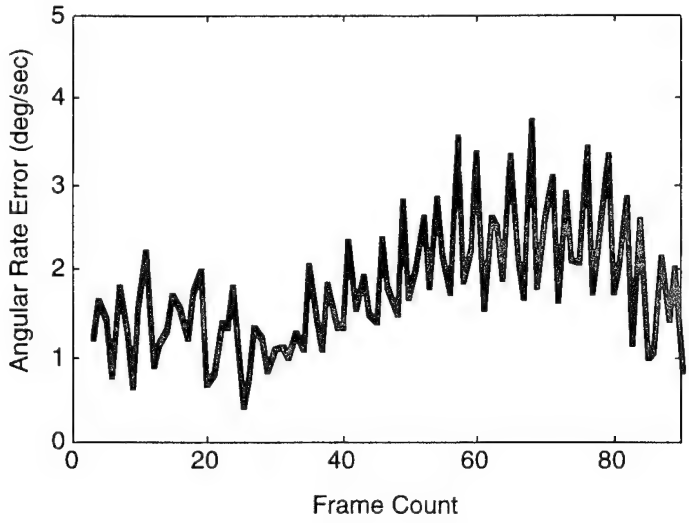


Figure 4.4-19b: Angular Rate Error History

RMS errors for the filtered impact time, aimpoint, and angular rate errors are 61.76%, 4.62⌋, and 2.15⌋/sec, respectively.

4.4.2.2 Testing Using Superpixellated Images

Figures 4.4-20 to 4.4-23 show estimation time histories for the helicopter impact time (altitude/speed), heading angle, flight path angle, and three individual angular velocity components, respectively using 8 x 8 kernel superpixellated images. As shown in figure 4.4-20, the impact time estimate now approaches the true impact time. The difference between the true and the filtered impact time is approximately 0.05 sec or 1.6 ft most of the time. As with the geometry-based estimator case, superpixillation significantly improves the accuracy of the estimated impact time.

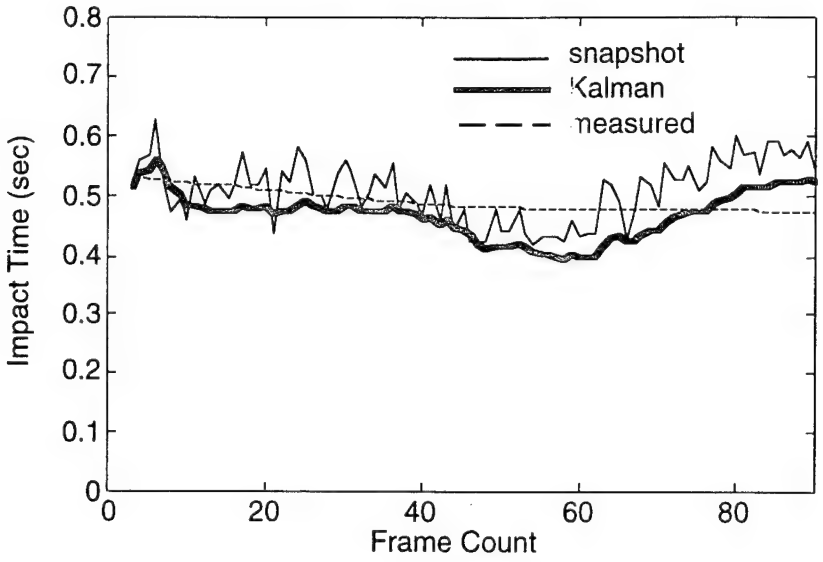


Figure 4.4-20: Impact Time Estimation History

Figures 4.4-21a and 4.4-21b illustrate time histories of the heading and flight path angle estimates, respectively. Superpixellation seems not affect the overall accuracy of the heading estimates. Note again the ability of the estimator in following the trend of true helicopter heading.

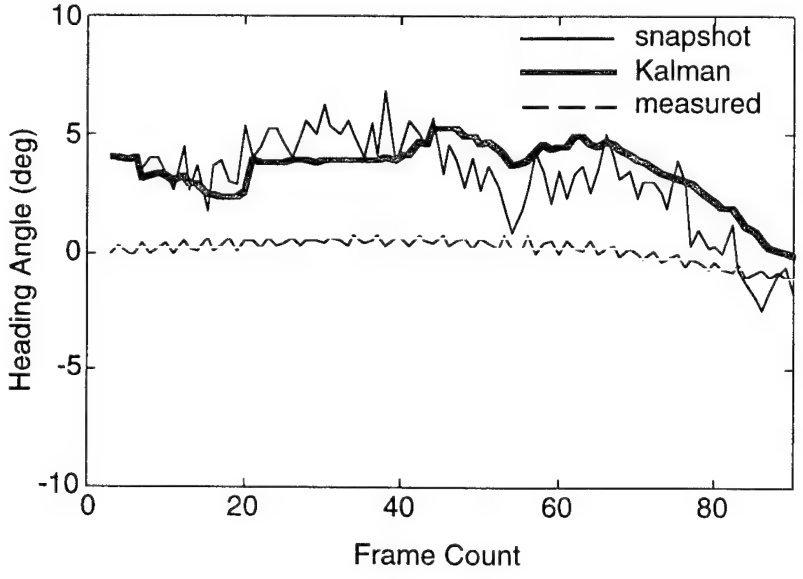


Figure 4.4-21a: Heading Angle Estimation History

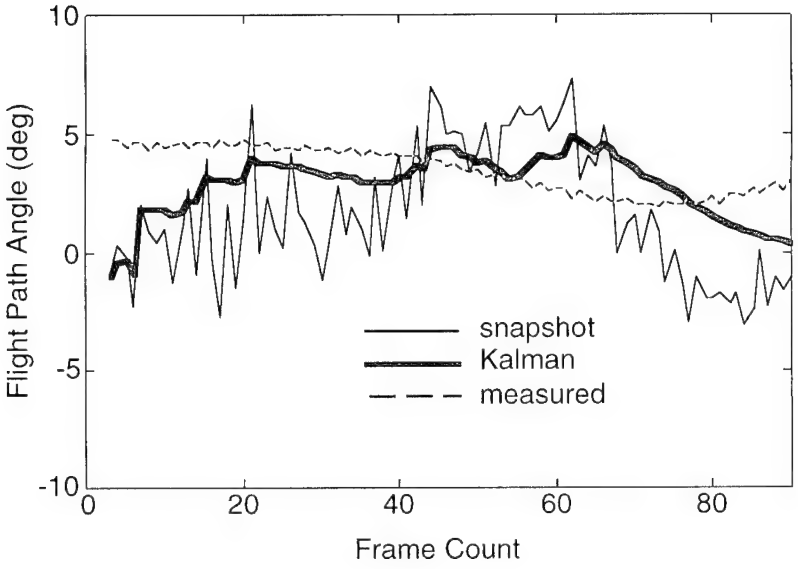


Figure 4.4-21b: Flight Path Angle Estimation History

Figures 4.4-22a, 4.4-22b, and 4.4-22c show histories of the three estimated angular velocity components. Superpixellation seems to reduce the ability of the estimator in closely following the trend of the measured angular velocity.

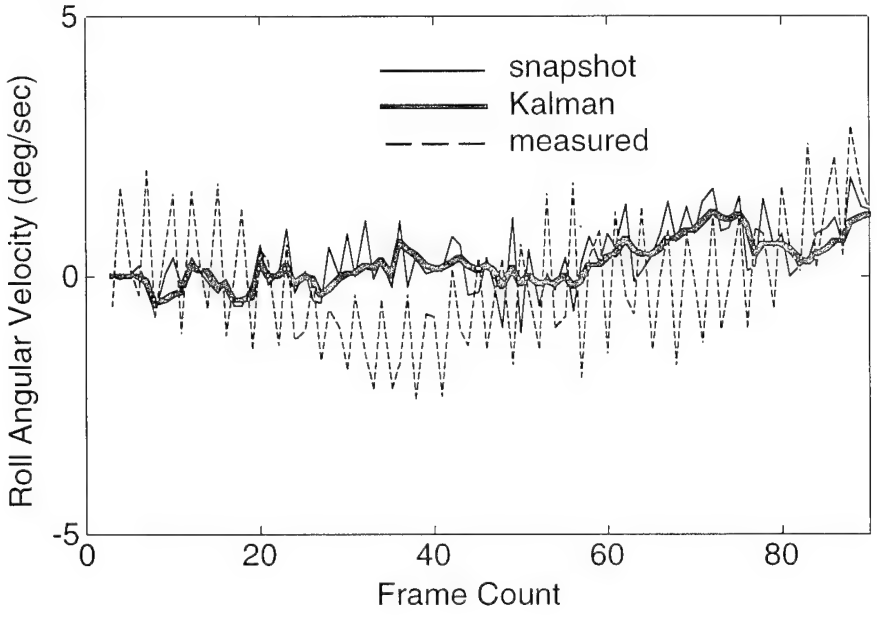


Figure 4.4-22a: Roll Angular Velocity Estimation History

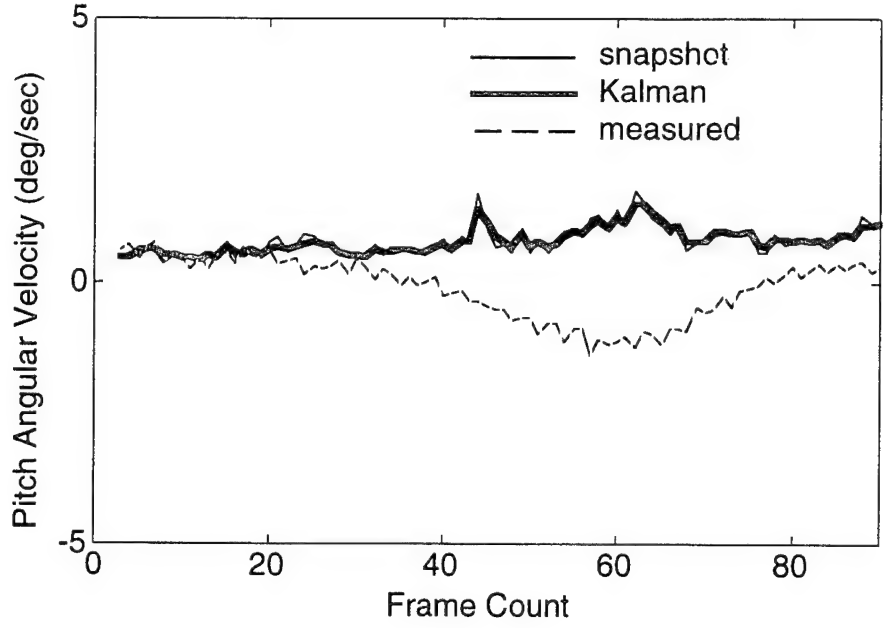


Figure 4.4-22b: Pitch Angular Velocity Estimation History

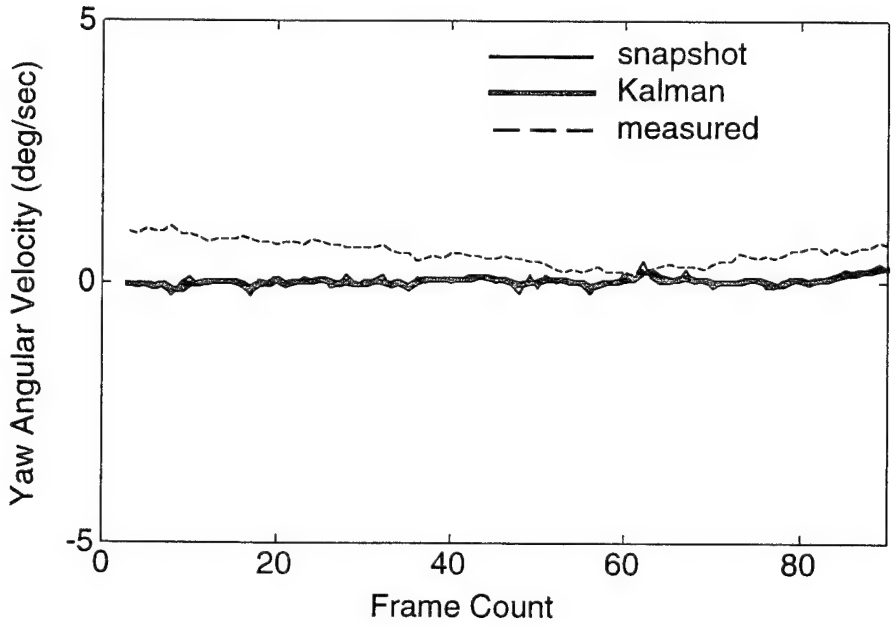


Figure 4.4-23c: Yaw Angular Velocity Estimation History

Figures 4.4-24, 4.4-25, and 4.4-26 show time histories of the impact time error, aimpoint error and angular rate error, respectively. The impact time error stays, most of the time, within a 10% error zone, the aimpoint error stays almost below 4.0°, and the angular rate error stays below 3.0°/sec.

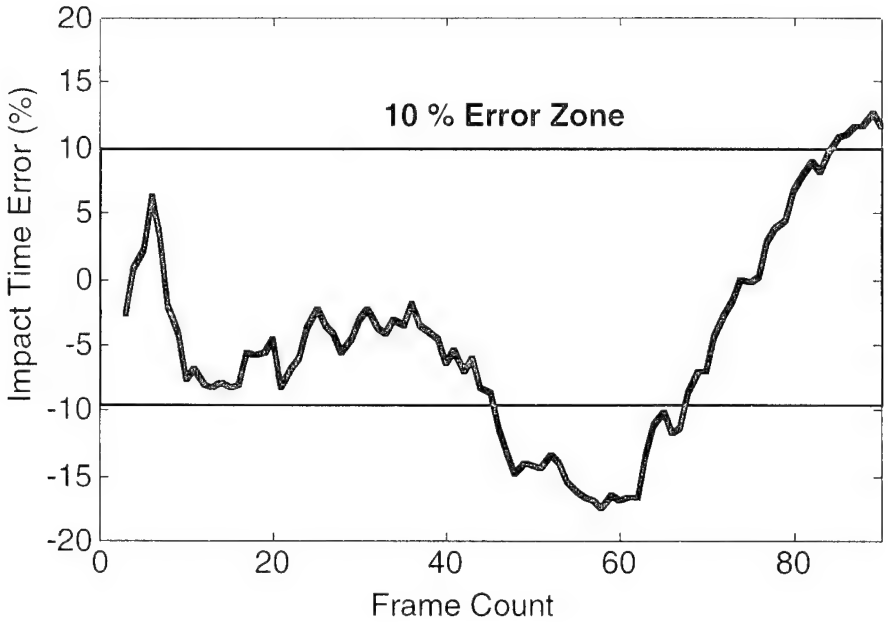


Figure 4.4-24: Impact Time Error History

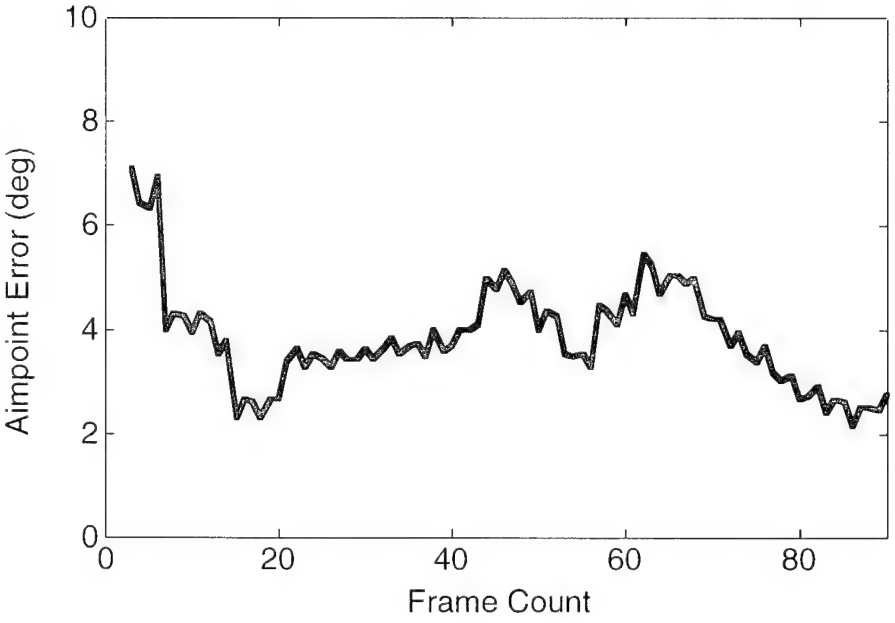


Figure 4.4-25: Aimpoint Error History

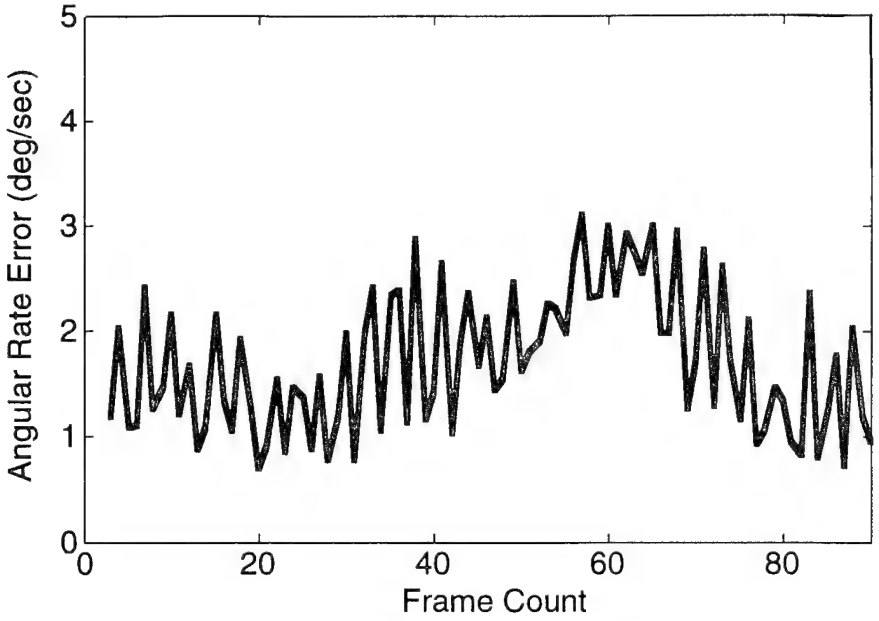


Figure 4.4-26: Angular Rate Error History

RMS errors for the filtered impact time, aimpoint, and angular rate errors are 10.02%, 3.93⌋, and 2.07⌋/sec, respectively.

4.5 Summary

We systematically tested and evaluated ATAS using the CGI and flight-recorded real-time frame grabbed terrain imagery. The evaluation results indicate that ATAS has a capability for accurate, efficient, and robust egomotion and terrain shape estimation under various terrain (flat and non-flat) and motion (constant and non-constant) conditions, when proper estimators (geometry-based or structure-based estimators) and proper pixellation sizes are selected. This capability is achieved through the development of a Kalman filter based approach to egomotion and terrain shape estimation. The approach employs a direct computation of the image gradient measurements (and the flow-field computations), a snapshot estimate of instantaneous egomotion and terrain shape using only the current image measurements, and a Kalman filtering process to integrate estimates over multiple image frames to reduce errors.

Using CGI, we evaluated ATAS for both flat and non-flat terrain images. With the flat terrain, ATAS had estimation errors of less than 2 degree in aimpoint, less than 3 degree/second in angular rate, and less than 5% in for altitude (impact time).

With the non-flat terrain images, we used a complex sequence that was generated by simulating a flight over the Yosemite valley. ATAS demonstrated an excellent ability to recover

the complicated Yosemite valley structure, whose complexity is far beyond the capability of any polynomial geometry terrain models (flat, quadratic, etc.). Moreover, ATAS had a less than 5 degree error in aimpoint, and a less than 1 degree/second error in angular rate, for these complicated non-flat terrain images.

For the flight-recorded and real-time grabbed terrain images, even though they were poorly "decorated", had very high noise-to-signal ratios, and the observer (airplane) was not undergoing constant motion, ATAS was still capable of generating accurate egomotion and terrain shape estimates. Specifically, it demonstrated errors of less than 5 degree in aimpoint, less than 2 degree/second in angular rate, and less than 10% in altitude (impact time). Moreover, ATAS showed an excellent ability in rapidly following and tracking the non-constant observer motion. Besides generating overall terrain shape estimates such as speed-scaled altitude, ATAS was also capable of recovering the shape of local objects (in this case, trucks present in the terrain images).

We also compared the advantages and disadvantages of the geometry-based and structure-based estimators, summarized in table 1.3-1.

The geometry-based estimator can only be applied to a terrain that can be approximated by a simple geometry model. Moreover, using a geometry-based model to approximate the entire terrain within the FOV makes it impossible to recover local structures hidden in the terrain image, such as isolated objects. In contrast, the structure-based approach has no limitation in its applicability. It can be applied to both simple and complicated terrain. Moreover, it can be used to recover the local structure of isolated objects in the imager FOV.

5 SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

This chapter concludes the report with a summary, conclusions, and recommendations for further development and demonstration.

5.1 Summary

Our Phase II technical approach to developing and demonstrating a prototype version of the passive sensor altitude and terrain awareness system (ATAS) was comprised of four tasks:

- Review of Current Approaches to Flow-Based Egomotion and Terrain Shape Estimation
- Enhancement and Development of a Prototype ATAS
- Development of a Test Bed for ATAS
- Demonstration and Evaluation of System Performance with CGI and flight-recorded imagery
- Recommendation of development path for a commercial product

We first **reviewed current approaches and evaluated their advantages and disadvantages**. A literature search was conducted specifically focusing on flow-field based estimation technologies. A review and summary of the most promising studies were conducted to identify capabilities and limitations of earlier flow-field based approaches.

We then **designed and implemented a prototype ground based ATAS** via specifying its overall architecture, its functional modules, and its software implementation. Our efforts were focused on the development of a robust, accurate, and efficient ATAS that can work in real-time. A Kalman filter based direct approach was developed that comprised of three modules: 1) an image processor; 2) a snapshot egomotion and terrain shape estimator; and 3) a Kalman filter providing smoothed estimates.

We also developed and implemented a test bed that has a capability for flat and non-flat computer terrain image generation. We defined and implemented various performance metrics for system evaluation.

Following implementation of the ground based ATAS and its test bed, we **demonstrated and evaluated system performance** via an extensive testing program. ATAS was tested using both CGI and real imagery, and using both the flat and non-flat terrain imagery. It was also tested against human experiments.

Finally, we **recommended a development path** for a commercial product, based on the

results of the evaluation effort. The development path includes an enhanced performance version of the Phase II design, and demonstration and performance evaluation over an extended CGI and real imagery database.

5.2 Conclusions

The primary result of this Phase II effort was the successful enhancement, implementation, and demonstration of the ATAS operation using both computer generated and real imagery. The major findings supporting this effort can be summarized as follows.

We developed an overall architecture of ATAS that comprises both the real-time image generation and real-time egomotion and terrain shape estimation. We developed and implemented a flat-terrain image generator that produces CGI frames simulating perspective images viewed by an observer (camera) flying over flat terrain decorated by arbitrary sinusoidal patterns. We also developed and implemented a rolling-terrain image generator that produces video (analog) imagery simulating images viewed by an observer flying over a sinusoidal rolling terrain decorated by sinusoidal patterns. In addition, from various sources we obtained other complex CGI terrain images used widely for computer vision algorithm testing. For real image generation, we obtained video imagery using a video camera fixed on a helicopter flying over real terrain. To convert the analog video imagery into digitized image frames, we implemented a frame grabber using hardware and software image grabbing tools hosted by a Macintosh Quadra 840 AV computer.

For egomotion and terrain shape estimation, we developed a three stage modular architecture comprised of an image processor; a snapshot egomotion and terrain shape estimator; and a Kalman filter, to provide smoother high-accuracy on-line estimates. This modular approach provided performance improvements and simplification for system implementation over the original design developed under the Phase I effort. First, it avoids the complicated interactions between low-level image processing and high-level motion state and terrain shape estimation. Second, separation of image processing, snapshot estimation, and Kalman filtering makes each problem simpler and its solution easier and more robust. Third, the system has a natural interface to incorporate external information for performance enhancement. For example, if the external information on observer motion state is available, it can then be easily incorporated into the Kalman filter without reconfiguration of the entire system. Finally, we believe that this modular architecture can provide the basis for a functional stands model of the human flow-field based vision process.

We developed two types of egomotion and terrain shape estimators to deal with different

kinds of terrain conditions: a geometry-based estimator and a structure-based estimator. The geometry-based estimator assumes that the terrain shape satisfies a polynomial geometry, and uses this model for terrain shape estimation. The structure based estimator does not assume *a priori* a terrain model, but extracts the hidden structural relationship among imaged terrain points using a multi-layer terrain structure model.

For static estimation in the snapshot estimator and dynamic estimation in the Kalman filter, we employed and implemented two state-of-the-art estimation algorithms: the Singular Value Decomposition (SVD) based Least Squares (LS), algorithm and the Square-Root Kalman filter algorithms. The application of these two numerically robust algorithms effectively overcomes any numerical instabilities that might be present in the estimation process.

Furthermore, the operation of ATAS needs specification of only a few meaningful parameters. For the geometry based approach, only four parameters need to be specified: the terrain model, the initial heading vector, the model standard deviations for motion state dynamic equations, and the discount factor for terrain impact time (range) map evolution; for the structure based approach, only the latter two parameters need to be specified. In contrast, many more parameters would be needed in a flow or feature matching based system, many of which need to be determined by trial and error.

We conducted a successful on-line demonstration of the ATAS ground-based prototype, driving the system with CGI and flight-recorded imagery, generating running estimates of altitude and attitude with respect to the overflown terrain. The evaluation results indicate that ATAS has a capability for accurate, efficient, and robust egomotion and terrain shape estimation over various terrain (flat and non-flat) and under different motion conditions, when the proper estimators (the geometry-based estimator or the structure-based estimator) and the proper pixellation sizes are selected.

Using CGI, we evaluated ATAS for both flat and non-flat terrain images. With the flat terrain, ATAS had estimation errors of less than 2 degree in aimpoint, less than 3 degree/second in angular rate, and less than 5% in for altitude (impact time).

With the non-flat terrain images, we used a complex sequence that was generated by simulating a flight over the Yosemite valley. ATAS demonstrated an excellent ability to recover the complicated Yosemite valley structure, whose complexity is far beyond the capability of any polynomial geometry terrain models (flat, quadratic, etc.). Moreover, ATAS had a less than 5 degree error in aimpoint, and a less than 1 degree/second error in angular rate, for these complicated non-flat terrain images.

For the flight-recorded and real-time grabbed terrain images, even though they were poorly "decorated", had very high noise-to-signal ratios, and the observer (airplane) was not undergoing constant motion, ATAS was still capable of generating accurate egomotion and terrain shape estimate. Specifically, it demonstrated errors less than 5 degree in aimpoint, less than 2 degree/second in angular rate, and less than 10% in altitude (impact time). Moreover, ATAS showed an excellent ability in rapidly following and tracking the non-constant observer motion. Besides generating overall terrain shape estimates such as speed-scaled altitude, ATAS was also capable of recovering the shape of local objects (in this case, trucks present in the terrain images).

We also compared the advantages and disadvantages of the geometry based and structure-based estimators, summarized in table 1.3-1.

The geometry-based estimator can only be applied to a terrain that can be approximated by a simple geometry model. Moreover, using a geometry-based model to approximate the entire terrain within the FOV makes it impossible to recover local structures hidden in the terrain image, such as isolated objects. In contrast, the structure-based approach has no limitation in its applicability. It can be applied to both simple and complicated terrain. Moreover, it can be used to recover the local structure of isolated objects in the imager FOV.

Large pixels can be used for image measurement computation in the geometry-based estimator. Large pixels not only reduce the computation load greatly, but also enhance overall estimation accuracy. The structure-based estimator, however, requires small pixels to properly define the terrain structure and thus incurs additional computational costs.

The geometry-based estimator has superior performance over the structure-based estimator when the actual terrain can be approximated by a simple geometry. In this case, the geometry-based model represents a true description of the terrain shape being imaged. The structure-based estimator, however, always has an approximate representation of the actual terrain, no matter how simple or complicated it actually is.

Using ATAS, we also analyzed several previous experimental studies of human egomotion perception. For the two reported on here, we used simple additive image noise to model the human's inability to register perfectly noise-free images, and we were able to show how this *perceptual noise* leads to corresponding inaccuracies in estimates of egomotion state. In particular, we showed how, within the model context, an assumed 10% to 15% image noise level leads directly to model-predicted egomotion aimpoint accuracy in the experimentally observed range of 1.5 to 2.5 degree. In addition, we showed how these aimpoint accuracy can be expected to co-vary with velocity-to-height ratios, which change by over a factor of 10, and confirmed

these trends with experimental data showing the same trends. Although more model validation is clearly needed, we feel that these initial matches between model and data are very encouraging.

Finally, we identified requirements for production system development and demonstration. Specifically, we recommended that the Phase II laboratory prototype serve as the basis for the development of a flight prototype system, to be used to demonstrate in-flight capabilities and system potential. A two-step development and demonstration program is envisioned: 1) development and demonstration of a flight prototype system configured for in-flight test and evaluation; and 2) specification of design requirements for a follow-on production system.

The Phase II findings demonstrated the overall capability of ATAS and established benchmark criteria for evaluating future systems. The effort demonstrated operation with off-the-shelf hardware, at performance levels that can significantly enhance critical on-board functions, including passive navigation, optically-based TF/TA, and FLIR-based target designation and weapons control.

5.3 Recommendations

On the basis of these Phase II prototype evaluations, we propose to develop a Phase III real-time flight system to meet specified performance, component compatibility, and budget constraints. A two step effort is proposed.

- The first step would be to transform ATAS from the current development host to the real-time host, to test the system operation on a terrain-board simulator, and to make design modification if deemed necessary.
- The second step would be to convert the ground based real-time host to a flight rated real-time host. and to conduct in-flight evaluations.

The first step would focus on the selection and configuration of the real-time host computer. The key system parameter in selecting the real-time host is its floating point processing power. Two types of real-time processing are required. The first type is real-time frame grabbing if a non-digital camera system were to be used for terrain image generation. The second type is the real-time image processing for the egomotion and terrain shape estimates. These have a total computational complexity of the order KN flops, where N is the total number of pixels and K is a coefficient accounting for the per pixel operations in image measurement computation, snapshot estimation, and Kalman filtering. As discussed earlier, the current development implementation has an associated K value of 100.

Table 5.3-1 illustrates the computational requirements for image processing of different sizes of image frames, assuming K=100, and assuming a 30 Hz frame rate.

Table 5.3-1: ATAS Computational Requirements

Number of Pixels	Total Flops
512 x 480 (standard NTSC)	24 Million
128 x 120 (4 kernel superpixellation)	1.5 Million
64 x 60 (8 kernel superpixellation)	0.4 Million

Based on these computational requirement analysis, we determine that an off-the-shelf Intel 586-based 120 MHz host computer can achieve the desired real-time requirements.

Figure 5.3-1 shows one potential architecture for a 586 PC-hosted ATAS real-time production system using an industry standard ISA bus to communicate with separate boards for frame grabbing and image processing. One particular advantage of the PC-based system is the large vendor base for all three types of boards. For the frame grabber and image co-processor boards, there are perhaps more than 10 vendors. We reviewed a number of available boards and found that board sets such as those provided by Imaging Technologies are particularly well-suited for the proposed application. For example, the Imaging Technologies VISIONplus-AT Advanced Frame Grabber Board operates in real-time with standard 30 Hz NTSC images, and captures them, digitizes them, and makes them available for further processing by downstream boards or the host. For follow-on co-processing of the grabbed image, including the superpixellation function, an image co-processing accelerator from Imaging Technologies can also be used. A candidate board is their 33 Mflop VISIONplus-AT Image Processing Accelerator (VIPA), which is connected to the front-end frame grabber via the high speed VISIONbus, to support real-time 30 Hz processing of 512 x 480 images, including image arithmetic, geometric transforms, and spatial filtering (convolution).

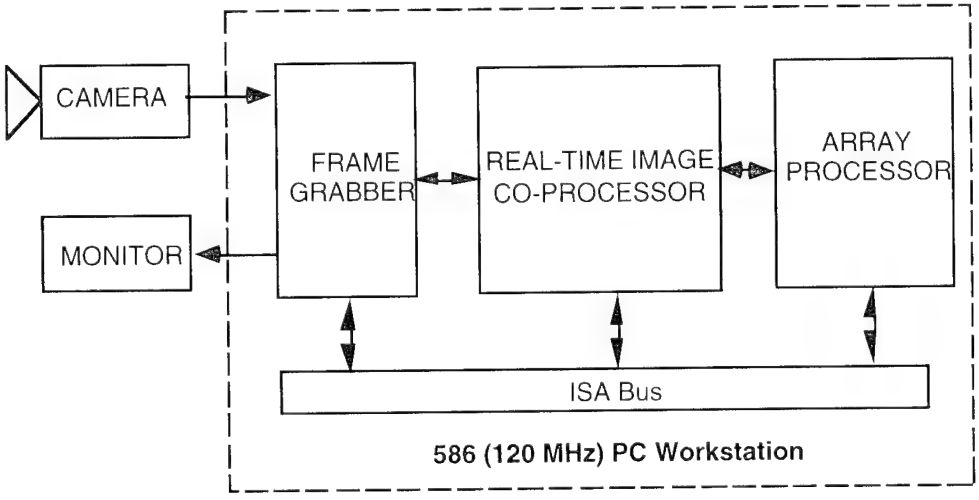


Figure 5.3-1: Architecture for Real-Time ATAS

With the image grabbed in real-time by the frame grabber board, the 120 MHz 586 PC host

would have enough power to conduct the egomotion and terrain shape estimation in real-time for superpixellated images. For 512 x 480 images, an additional array processor might be needed for the real-time processing, as shown in 5.3-1.

Table 5.3-2 shows the estimated costs for the proposed real-time system. Note that the array processor board may not be necessary. We have thus indicated it as optional.

Table 5.3-2: Estimated Costs for the Proposed Real-Time ATAS

HARDWARE	PRICE ($)	TOTAL COST ($)
120 MHz 586-based PC with 32 Mb RAM 1 Gb HD Quad Speed CD-ROM	3,100	
Frame Grabber Board IT Vision Plus AT and software	3,000	
Image CO processor Board IT Vision Plus AT and software	7,000	13,100
Array Processor Board (optional)	11,000	24, 100

The second step would focus on the conversion of the ground-based real time ATAS to a flight rated real time system. Four subtasks are involved: 1) a specification and configuration of the 586 PC host to meet MIL-SPEC requirements; 2) an integration of the host with the on-board imaging sensor; 3) porting and checkout of the flight system; and 4) flight testing to determine the system performance.

For the Phase III flight system (FS) of ATAS, we propose a three-bus architecture, as illustrated in figure 5.3-2. The DoD MIL-STD-1553B Data Bus for avionics is shown as the top-most bus, interlinking the ATAS real-time production system (shown in the dashed box) with the other on-board systems. The Industrial Standard Architecture (ISA) bus is shown as the middle bus, and serves to interconnect several required boards on a PC host. The VISIONBUS, a proprietary bus by Imaging Technologies, Inc., is shown as the bottom-most bus, and serves to interconnect image processing boards requiring very high data bandwidths.

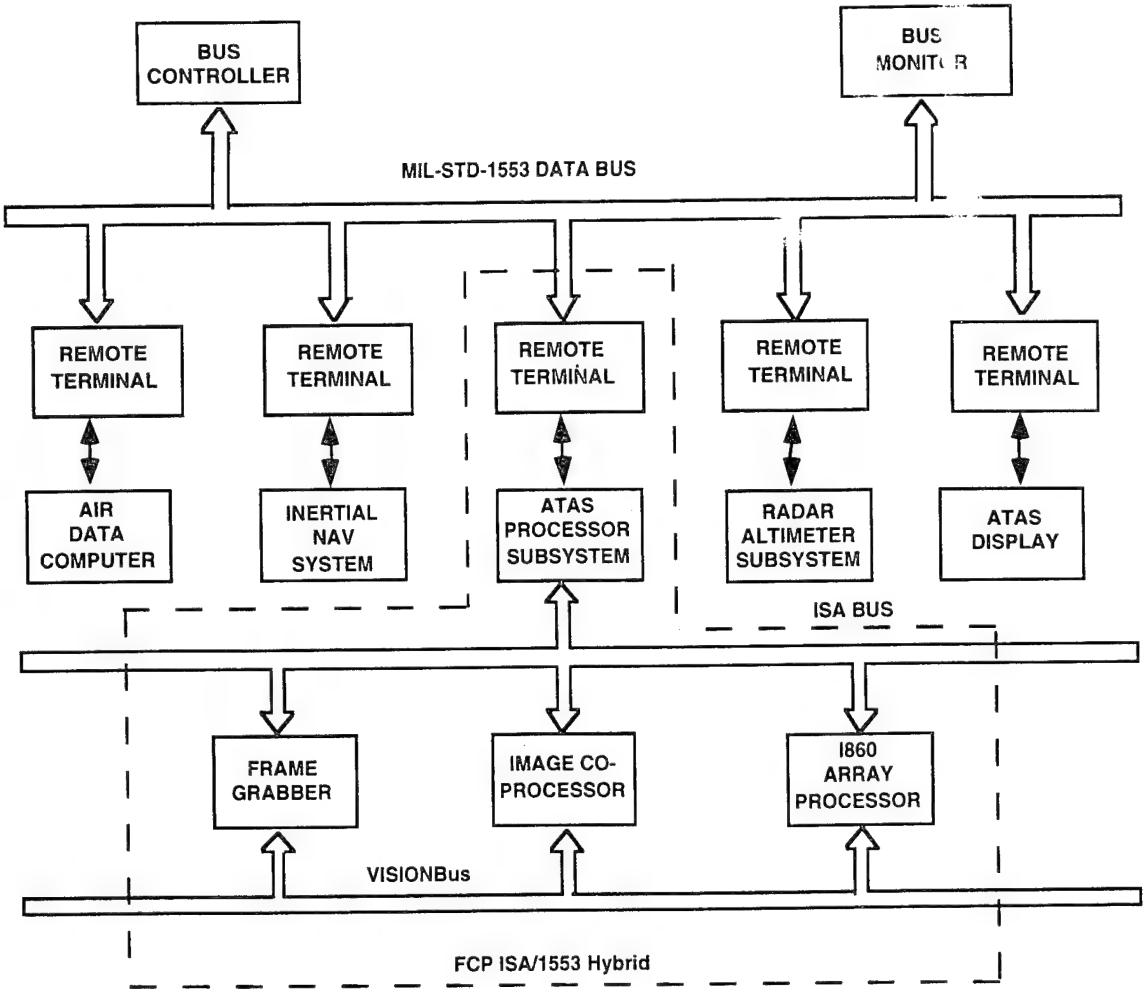


Figure 5.3-2: Proposed Architecture for ATAS Flight Prototype System (FPS)

The **1553B data bus** interlinks the ATAS processor with the several remote terminals serving other on-board systems. The purpose of this bus is to provide access to on-board measurements and system parameters to validate ATAS operation and estimation accuracy. Specifically shown is the air data computer (ADC) which provides barometric altitude, the inertial navigation system (INS) which provides vehicle altitude, location, and ground speed, the RADAR altimeter subsystem which provides actual altitude for verification purposes, and an ATAS display, for optional display of ATAS-generated terrain profiles. Although not shown on the diagram, it is anticipated that a terrain following radar (TFR) system, if available, would also be accessed, to allow for the direct comparison of TFR-generated terrain shape estimates with those generated by the ATAS processor.

The 1553B data bus itself is defined formally by MIL-STD-1553B (1978), which establishes the requirements for digital data buses on military aircraft. The standard specifies:

1. Types of terminals: bus controller, bus monitor, and remote terminal
2. Bus protocol, including message formats and word structure
3. Hardware performance specifications, such as characteristic impedance and connection requirements.

The proposed FS will interface with the data bus via a Remote Terminal, which will respond only to commands specifically addressed to it. We plan to embed the remote terminal within the processor subsystem, for maximum interfacing autonomy. The proposed system will use the bus protocol detailed by the 1553 message formats, or *information transfer formats*. The **ISA bus** shown in figure 5.3-2 serves as the backbone of the ATAS FPS, outlined by the dashed box in the figure. Currently under development by the Crew Systems Integration Branch of the Armstrong Laboratory (AL/CFHI), as part of the Flight Computer Project (FCP), is an ISA-based computer which will directly interface to the 1553B bus as a remote terminal (Turner, 1992). We propose its use during the Phase II effort, as the host processor for the ATAS FPS.

The FCP computer consists of an Intel 80586 processor operating at 120 MHz, 32 Mb RAM, one Gig hard drive, a Quad Speed CD-ROM drive, and serial and parallel ports. The interface between the 1553 and ISA buses is a 1553 Bus Twin-Ax Connector, a BN73 Tee Connector, and a BJ77 Backpanel Connector. Available within the system box are three (16-bit) ISA slots, which we propose to use for the image processing cards, as shown in the figure.

The FCP computer holds another advantage besides the hybrid ISA/1553 bus structure: fully integrated into the computer is a Global Positioning System (GPS), which can provide worldwide navigation coverage to an accuracy of a few meters. The satellite-based system operates via transmission of precisely timed signals to a ground-based GPS receiver. Using known satellite positions, and triangulation over multiple satellites, the GPS receiver computes the unknown signal transmit delay and the receiver's unknown location. This is then made available to the host system for further processing. Six orbital planes with three satellites per plane provide round-the-clock coverage. GPS position could be used to generate local terrain elevation during flight tests, if the overflights are made over surveyed regions, and if the survey is available in digital format. Combining a map-based elevation profile with the instantaneous measurement available from a radar altimeter would then allow us to accurately compute actual elevation above the terrain, for later off-line verification of system operation.

Two options for the display hardware are possible. First, the information to be displayed could be sent via the 1553B data bus to an ATAS display connected to the bus. This will be an attractive option if the flight prototype is equipped with a general purpose multi-function CRT display. The second option is to display the information on a dedicated ISA-based video display

which is part of the FCP computer. The main advantage of this option is that there is no need to interface with existing flight hardware, and all display development could be accomplished on the ground-based demonstrator, since the same ISA-based display hardware would be available.

We propose to evaluate performance of the Phase III FS using recorded synthetic and real-image sequences, and via in-flight testing.

The recorded image sequences will include all those used in the GBD evaluation effort, including: Phase II image sequences, CGI sequences recorded for flat and rolling terrain, high-fidelity visual flight simulator image sequences, and flight recorded sequences available through the NASA Ames Research centers. Direct comparison with the previous GBD test results will serve to validate operation of the FS, prior to flight testing. Demonstration of in-line real-time FS operation will then proceed, with the flight demonstration portion of the effort.

A variety of aircraft are under consideration for the Phase III in-flight demonstration. These include the DC-3 LLTV/FLIR-equipped imager testbed maintained by Texas Instruments, Inc., an F-16 operated by General Dynamics providing equipped F-16 operated by the 6516th Test Squadron at Edwards AFB, several potential platforms operated by the 4950th Test Wing, and the NASA Shuttle Telescope testbed. In addition, a variety of private non-government platforms could be readily configured for the ATAS flight demonstration; several candidates are currently under evaluation.

The flight demonstration will involve an initial effort in installation and checkout in the host aircraft, in-flight demonstration of a range of maneuvers and terrain conditions chosen to test the system to its limits, and post-flight analysis of system performance. We recognize that demonstration of in-flight FPS operation will present a significant challenge, primarily because the general difficulty in conducting flight simulations of any type. However, we plan to conduct extensive pre-flight simulation of ATAS operation, both under the FS design effort, and under the GBD effort. This will be aimed at defining a minimal flight test sequence which will best demonstrate ATAS capabilities and limitations in flight.

6. References

Anandan, P. (1989). "A Computational Framework and an Algorithm for the Measurement of Visual Motion". *International Journal of Computer Vision*, 2, 283-310.

Anderson, B. D. O., & Moore, J. B. (1979). *Optimal Filtering*. Englewood Cliffs, NJ: Prentice Hall.

Barron, J. L., Fleet, D. J., & Beauchemin, S. S. (1994). "Systems and Experiment, Performance of Optical Flow Techniques". *International Journal of Computer Vision*, 12, 43-77.

Barron, J. L., Jepson, A. D., & Tsotsos, J. K. (1990). "The Feasibility of Motion and Structure from Noisy Time-Varying Image Velocity Information". *Int. Journal of Computer Vision*, 5, 239-269.

Broida, T. J., & Chellappa, R. (1986). "Estimation of Object Motion Parameters from Noisy Images". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8, 90-99.

Burden, R. L., & Faires, J. D. (1989). *Numerical Analysis*. Boston, MA: PWS-KENT Publishing Company.

Fleet, D. (1992). *Measurement of Image Velocity*. Norwell, MA: Kluwer Academic Publishers.

Fleet, D. J., & Jepson, A. D. (1990). "Computation of Component Image Velocity from Local Phase Information". *International Journal of Computer Vision*, 5, 77-104.

Grunwald, A. J., & Kohn, S. (1993). "Flight-Path Estimation in Passive Low-Altitude Flight by Visual Cues". *Journal of Guidance, Control, and Dynamics*, 10(2, March/April), 363-370.

Heeger, D. J. (1987, August). "Model for the Extraction of Image Flow". *Journal of the Optical Society of American A*, 4, 1455-1471.

Heeger, D. J., & Jepson, A. D. (1992). "Subspace Methods for Recovering Rihid Motion I: Algorithm and Implementation". *International Journal of Computer Vision*, 7(2), 95-117.

Horn, B. K. P., & Schunck, B. G. (1981). "Determining Optical Flow". *Artificial Intelliegence*, 17, 185-204.

Kamgar-Parsi, B., & Kamgar-Parsi, B. (1989). "Evaluatoin of Quantization Error in Computer Vision". *IEEE Transactiopns on Pattern Analysis and Machine Intelligence*, 11(9, September), 929-940.

Little, J. J., & Verri, A. (1989). "Analysis of Differential and Matching Methods for Optical Flow". *IEEE Workshop on Visual Motion*, Irvine, CA.

Lucas, B., & Kanade, T. (1981). "An Iterative Image Registration Technique and Application to Stereo Vision". *Proc. DARPA Image Understanding Workshop*.

Matthies, L., Szelisky, R., & Kanada, T. (1989). "Kalman Filter Based Algorithms for Estimating Depth from Image Sequences". *International Journal of Computer Vision*, 2, 209-237.

Maybeck, P. S. (1979). *Stochastic Models, Estimation and Controls*. New York, NY: Academic Press.

Nagel, H. H. (1983). "Displacement Vectors Derived from Second-Order Intensity Variations in Image Sequences". *Computer Graphic Image Process*, 21, 85-117.

Negahdaripour, S. (1986). *Direct Passage Navigation*. Ph.D., MIT.

Negahdaripour, S., & Horn, B. K. P. (1989). "A Direct Method for Locating the Focus of Expansion". *Computer Vision Graphic Image Process*, 46(3).

Negahdaripour, S., & Lee, S. (1992). "Motion Recovery from Image Sequences Using Only First Order Optical Flow Information". *International Journal of Computer Vision*, 9(3), 163-184.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical Recipes in C*: Cambridge University Press.

Singh, A. (1992). *Optical Flow Computation: A Unified Perspective*: IEEE Computer Society Press.

Smith, P. N. (1990). *NASA Image Data Base User's Guide*. (Vol. Version 1.0). Moffett Field, CA: NASA Ames Research Center.

Sridhar, B., & Chatterji, G. B. (1994). "Vision-Based Obstacle Detection and Grouping for Helicopter Guidance". *Journal of Guidance, Control and Dynamics*, 17(5), 908-915.

Turner. (1992). "LANTIRN Test Group, Personal Communcation", .

Warren, R. (1976). "The Perception of Egomotion". *J. Exp. Psych: Human Perception and Performance*, 2, 448-456.

Wu, J. J., Rink, R. E., Caelli, T. M., & Gourishankar, V. G. (1989). "Recovery of the 3-D Location and Motion of a Rigid Object Through Camera Image (An Extended Kalman Filter Approach)". *Int. Journal of Computer Vision*, 3, 373-394.

Zacharias, G. L., Caglayan, A. K., & Sinacori, J. B. (1985). "A Model for Visual Flow-Field Cueing and Self-Motion Estimation". *IEEE Trans on Systems, Man, & Cybernetics*, 15(3), 385-389.

Zacharias, G. L., & Levison, W. H. (1980). *Development of a Model for the Use of Extra-Cockpit Visual Cues* (4562): Bolt Beranek and Newman Inc. (December).

Zacharias, G. L., Miao, A. X., & Riley, E. W. (1992). *Passive Sensor Altitude and Terrain Awareness System* (Final Report R91071): Charles River Analytics (February 1992).

Zacharias, G. L., Miao, A. X., & Warren. (1995). "Multistage Integration Model for Human Egomotion Perception". *Journal of Guidance, Control & Dynamics*, to be published.

Appendix A: Image Gradient Error Analysis

We first consider image gradient errors induced by spatial and temporal quantization. From the two-point ceneter formulas (3.1-1a,b, c), we can see that spatial quantization would affect the spatial gradient error through the spatial steps Δx_i and Δy_i, and the temporal quantization would affect the temporal gradient error through the temporal step Δt. We start with the analysis of spatial gradient error. Let us assume that the same pixel size is adopted in x and y directions, i.e., $\Delta x_i = \Delta y_i \equiv \Delta_i$. When the three point formula is used to estimate the gradient of a continuous function f(x), the estimation error is given by

$$\Delta[\hat{f}'(x)] = -\frac{\Delta^2}{6} f'''(x + \xi) \tag{A-1}$$

where Δ is the differential step and ξ lies between $-\Delta$ and Δ (Burden & Faires, 1989). Using (A-1), we obtain the spatial gradient error

$$\Delta_q(\hat{\mathbf{g}}_i) = -\frac{\Delta_i^2}{6}[I'''_{x^3}(x_i + \xi, y_i, t), I'''_{y^3}(x_i, y_i + \zeta, t)]^T \tag{A-2}$$

where $-\Delta_i \le \xi \le \Delta_i$ and $-\Delta_i \le \zeta \le \Delta_i$.

We continue to determine the temporal gradient error. Let $\mathbf{v}_i = (\dot{x}_i, \dot{y}_i)$ be the image flow at pixel i. In the observer's frame of reference, the temporal intensity change at pixel i can be regarded as a result of the image intensity profile moving with the velocity $-\mathbf{v}_i$. In other words, the image intensity function can also be represented by

$$I(x_i, y_i, t) = I(x_i - \dot{x}_i t, y_i - \dot{y}_i t) \tag{A-3}$$

The temporal gradient error induced by temporal quantization is then given by

$$\Delta_q(\hat{I}_{it}) = -\frac{(\Delta t)^2}{6} I'''_{t^3}(x_i - \dot{x}_i \iota, y_i - \dot{y}_i \iota)$$

$$= \frac{(\Delta t)^2}{6} \sum_{n=0}^{3} C_3^n I'''_{x^{3-n} y^n}(x_i - \dot{x}_i \iota, y_i - \dot{y}_i \iota) \dot{x}_i^{3-n} \dot{y}_i^n \tag{A-4a}$$

where C_3^n is the combinatorial coefficient,

$$I'''_{x^{3-n} y^n}(x_i - \dot{x}_i \iota, y_i - \dot{y}_i \iota) = \frac{\partial^3 I(x_i - \dot{x}_i \iota, y_i - \dot{y}_i \iota)}{\partial x^{3-n} \partial y^n} \tag{A-4b}$$

and $-\Delta t \le \iota \le \Delta t$.

The actual amount of error in either spatial or temporal gradient depends on the value of ξ, ζ, or ι. For small Δ_i and Δt, we assume that the probability of either ξ (ζ) or ι being at a certain place within the interval $[-\Delta_i, \Delta_i]$ or $[-\Delta t, \Delta t]$ follows a uniform probability distribution. The expected spatial and temporal gradient errors at pixel i are then obtained as:

$$\bar{\Delta}_q(\hat{\mathbf{g}}_i) = -\frac{\Delta_i^2}{6}[I'''_{x^3}(x_i, y_i), I'''_{y^3}(x_i, y_i)]^T$$

$$\bar{\Delta}_q(\hat{I}_{it}) = \frac{(\Delta t)^2}{6}\sum_{n=0}^{3} C_3^n I'''_{x^{3-n}y^n}(x_i, y_i)\dot{x}_i^{3-n}\dot{y}_i^n \qquad \text{(A-5a, b)}$$

Under the same probabilistic view of the image gradient error, we can also estimate the image gradient error variances. Specifically, the variance for the x-direction spatial gradient is estimated by

$$\sigma_q^2(\hat{I}_{ix}) = E[\Delta_q(I_{ix}) - \bar{\Delta}_q(I_{ix})]^2$$

$$= \frac{\Delta_i^4}{36}E[I'''_{x^3}(x_i + \xi, y_i) - I'''_{x^3}(x_i, y_i)]^2$$

$$\approx \frac{\Delta_i^6[I''''_{x^4}(x_i, y_i)]^2}{108} \qquad \text{(A-6a)}$$

Similarly, we have the variances for the y-direction spatial gradient and temporal gradient as

$$\sigma_q^2(\hat{I}_{iy}) \approx \frac{\Delta_i^6[I''''_{y^4}(x_i, y_i)]^2}{108} \qquad \text{(A-6b)}$$

$$\sigma_q^2(\hat{I}_{it}) \approx \frac{(\Delta t)^6[\sum_{n=0}^{4} C_4^n I''''_{x^{4-n}y^n}(x_i, y_i)\dot{x}_i^{4-n}\dot{y}_i^n]^2}{108} \qquad \text{(A-6c)}$$

We now consider the image gradient error induced by intensity quantization. Let $\Delta_g[I(x_i, y_i, t)]$ represent the intensity quantization error. We can directly compute the image gradient error from the three point formula:

$$\Delta_g(\hat{I}_{ix}) = \frac{\Delta_g^x(I_i)}{2\Delta_i}, \Delta_g(\hat{I}_{iy}) = \frac{\Delta_g^y(I_i)}{2\Delta_i}, \text{ and } \Delta_g(\hat{I}_{it}) = \frac{\Delta_g^t(I_i)}{2\Delta t}$$

where

$$\Delta_g^x(I_i) = \Delta_g[I(x_i + \Delta x, y_i, t - \Delta t)] - \Delta_g[I(x_i - \Delta x, y_i, t - \Delta t)]$$

$$\Delta_g^y(I_i) = \Delta_g[I(x_i, y_i + \Delta y, t - \Delta t)] - \Delta_g[I(x_i, y_i - \Delta y, t - \Delta t)]$$

$$\Delta_g^t(I_i) = \Delta_g[I(x_i, y_i, t)] - \Delta_g[I(x_i, y_i, t - 2\Delta t)] \qquad \text{(A-7a, b, c, d)}$$

Let G denote the grey levels used to represent image intensity. Assuming that the intensity quantization error follows a uniform distribution within the quantization interval $[-\frac{1}{2G}, \frac{1}{2G}]$, we obtain the average errors

$$\bar{\Delta}_g(\hat{I}_{ix}) = \bar{\Delta}_g(\hat{I}_{iy}) = \bar{\Delta}_g(\hat{I}_{it}) = 0 \qquad \text{(A-8)}$$

and the variances:

$$\sigma_g^2(\hat{I}_{ix}) = \sigma_g^2(\hat{I}_{iy}) = \frac{1}{24G^2\Delta_i^2} \text{ and } \sigma_g^2(\hat{I}_{it}) = \frac{1}{24G^2(\Delta t)^2} \qquad \text{(A-9)}$$

using the computation formulas given by Kamgar-Parsi (1989).

Appendix B: Optimal Assignment Algorithm

The optimal assignment algorithm is derived using Sherman-Morrison formula (Press et al., 1992) to compute the inverse of a matrix when one of its rows or columns is changed. For matrix **A** and some column vectors **u** and **v**, the Sherman-Morrison formula is given as

$$\mathbf{A} + \mathbf{u}\mathbf{v}^T)^{-1} = \mathbf{A}^{-1} - \frac{(\mathbf{A}^{-1}\mathbf{u})(\mathbf{v}\mathbf{A}^{-1})}{1 + \mathbf{v}^T\mathbf{u}} \tag{B-1}$$

We now develop an iterative algorithm that optimally assigns N terrain points among M structure layers, one point at each iteration. Let n be the current iteration, a_i^n be the point i's current assignment, and $e(\{a_i^n\})$ be the current projection length. At iteration n+1, we change the point i's assignment to $a_i^{n+1} = 1, 2, ..., M$. Denote

$$\mathbf{B}^n \equiv \mathbf{H}^{nT}\mathbf{H}^n,\ \mu^n \equiv \mathbf{h}_i^T(\mathbf{B}^n)^{-1}\mathbf{h}_i,\ \gamma_i^n = \mathbf{b}^n(\mathbf{B}^n)^{-1}\mathbf{h}_i \tag{B-2}$$

Using the Sherman-Morrison formula to compute the matrix inverse $[\mathbf{B}^{nT} + ((a_i^{(n+1)})^2 - (a_i^n)^2)\mathbf{h}_i\mathbf{h}_i^T]^{-1}$, we have the projection length at iteration n+1

$$\begin{aligned} e(\{a_i^{n+1}\}) &\equiv [\mathbf{b}^{nT} + b_{it}(a_i^{n+1} - a_i^n)\mathbf{h}_i^T][\mathbf{B}^{nT} + ((a_i^{(n+1)})^2 - (a_i^n)^2)\mathbf{h}_i\mathbf{h}_i^T]^{-1}[b_{it}(a_i^{n+1} - a_i^n)\mathbf{h}_i + \mathbf{b}^n] \\ &= e(\{a_i^n\}) + \frac{\mu_i^n b_{it}^2(a_i^{n+1} - a_i^n)^2 + 2\gamma_i^n b_{it}(a_i^{n+1} - a_i^n) - (\gamma_i^n)^2[(a_i^{(n+1)})^2 - (a_i^n)^2]}{1 + \mu_i^n(a_i^{(n+1)})^2 - (a_i^n)^2]} \\ &= e(\{a_i^n\}) + \frac{[\mu_i^n b_{it}^2 - (\gamma_i^n)^2](a_i^{n+1} - a_i^n)^2 + 2(b_{it} - a_i^n\gamma_i^n)\gamma_i^n(a_i^{n+1} - a_i^n)}{\mu_i^n(a_i^{n+1} - a_i^n)^2 + 2\mu_i^n a_i^n(a_i^{n+1} - a_i^n) + 1} \end{aligned} \tag{B-3}$$

We want to select an assignment a_i^{k+1} that maximizes $e(\{a_i^{k+1}\})$. To do so, we find the assignment a_i^{k+1} that has zero gradient

$$\begin{aligned} 0 &\equiv \frac{\partial e(\{a_i^{k+1}\})}{\partial a_i^{k+1}} \\ &= 2\frac{[a_i^k\mu_i^k b_i - \gamma_i^k]\mu_i^k b_{it}(a_i^{k+1} - a_i^k)^2 + [\mu_i^k b_{it}^2 - (\gamma_i^k)^2](a_i^{k+1} - a_i^k) + [b_{it} - a_i^k\gamma_i^k]\gamma_i^k}{[\mu_i^k(a_i^{k+1} - a_i^k)^2 + 2\mu_i^k a_i^k(a_i^{k+1} - a_i^k) + 1]^2} \end{aligned} \tag{B-4}$$

Two zero gradient solutions can be found by solving the quadratic function

$$[a_i^k\mu_i^k b_i - \gamma_i^k]\mu_i^k b_{it}(a_i^{k+1} - a_i^k)^2 + [\mu_i^k b_{it}^2 - (\gamma_i^k)^2](a_i^{k+1} - a_i^k) + [b_{it} - a_i^k\gamma_i^k]\gamma_i^k = 0 \tag{B-5}$$

Figure B-1 illustrates the geometry of $e(\{a_i^{k+1}\})$ function that has two zero gradient points. From the figure and basic function optimization theory, we can see that the optimal assignment of a_i^{k+1} is to assign a_i^{k+1} to one of the zero gradient layer that increases the projection length.

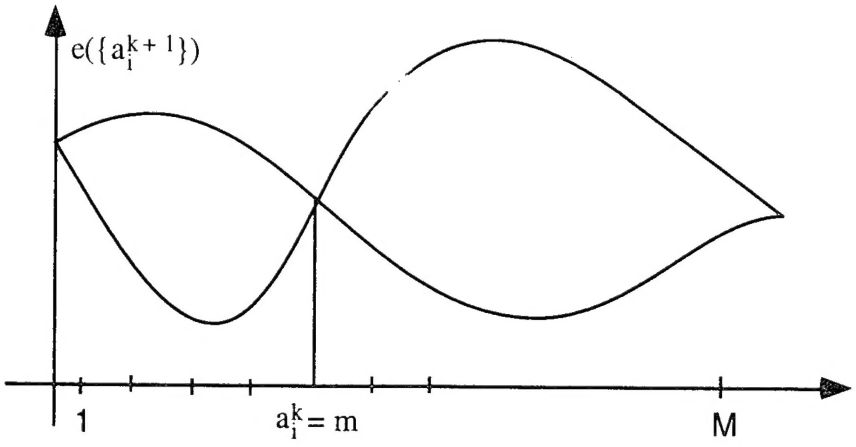


Figure B-1: Geometry of Projection Length Function

Appendix C: Motion State and Terrain Shape Representation in a Selected Coordinate System

The egomotion and terrain shape estimation system estimates the motion state and terrain shape relative to the screen coordinate system. The estimated motion state and terrain shape can be represented in any observer-fixed coordinate system via a coordinate system transformation, as shown in figure C-1.

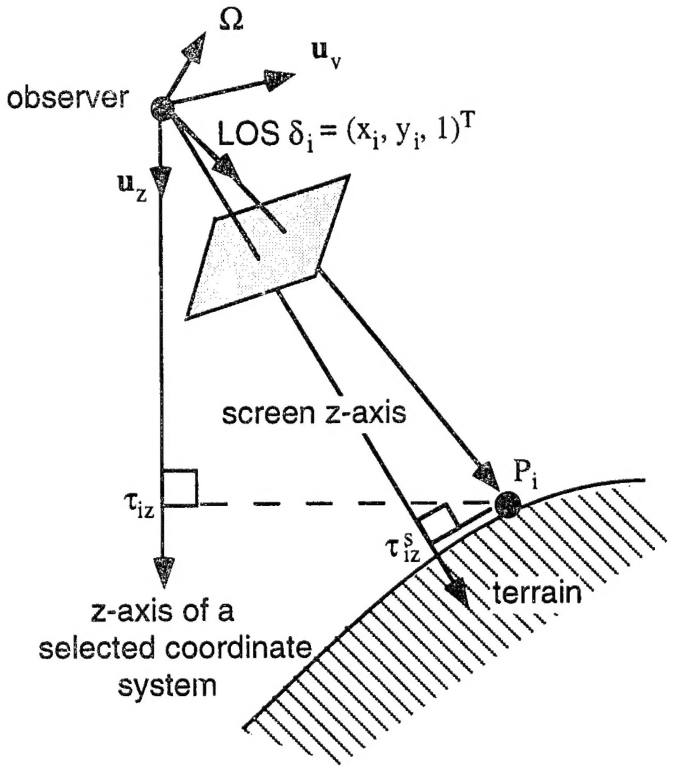


Figure C-1: Projection Transform between Coordinate Systems

Figure C-1 shows the z-axis coordinates τ^s_{iz} and τ_{iz} of a terrain point P_i in the screen and a selected observer-fixed coordinate system, such as body coordinate system. Let $\mathbf{u}_z = (0, 0, 1)^T$ be the z-axis unit vector of the selected coordinate system, and $\mathbf{T}_{sa}$ be the transformation matrix between the screen and the selected coordinate system. The relationship between τ^s_{iz} and τ_{iz} is given by the projection equation

$$\tau_{iz} = (\mathbf{u}_z \circ \mathbf{T}_{sa}\delta_i)\tau^s_{iz} \tag{C-1}$$

Let Ω^s and $\mathbf{u}^s_v$ be the angular velocity and heading vectors in the screen coordinate system, respectively. Their representations in the selected coordinate system are given by

$$\Omega = \mathbf{T}_{sa}\Omega^s \tag{C-2}$$

$$\mathbb{u}_v = \mathbb{T}_{sa}\mathbb{u}_v^s \qquad \text{(C-3)}$$

Furthermore, the operation of ATAS needs specification of only a few meaningful parameters. For the geometry-based approach, only four parameters need to be specified: the terrain model, the initial heading vector, the model standard deviastions for motion state dynamic equations, and the discount factor for terrain impact time (range) map evolution; for the structure-based approach, only the latter two parameters need to be specified. In contrast, many more parameters would be needed in a flow or feature matching based system, many of which need to be determined by trial and error.

U. S. Government Printing Office 1996 750-071/00178